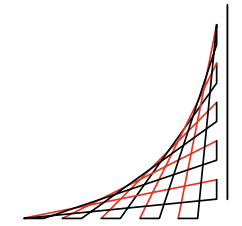
Introducción a la Computación Bioinspirada: Diseño de asignatura y creación de material de aprendizaje

Proyecto Dirigido
Ingeniería de Sistemas

Autores:
Juan Camilo Rojas Ortiz
Juan Camilo Angel Hernandez
Directora:
María Irma Díaz Rozo



Decanatura de Ingeniería de Sistemas
Escuela Colombiana de Ingeniería Julio Garavito
Bogotá, Colombia
Julio 2020

Resumen

Este trabajo corresponde al diseño y construcción de una asignatura universitaria la cual lleva como nombre *Introducción a la Computación Bioinspirada*, aquí se realizó el diseño de la asignatura y la creación del material de aprendizaje. El desarrollo de este proyecto se encuentra dentro del marco de la computación natural, la computación natural es un campo de investigación que tiene como propósito comprender los procesos inteligentes que tienen lugar en la naturaleza y desarrollar modelos computacionales inspirados en este comportamiento. Este campo se divide en tres áreas:

- Computación inspirada por la naturaleza.
- Simulación de la naturaleza por medio de la computación.
- Computación con materiales naturales.

Para la ingeniería y por consiguiente para nosotros, la tendencia de mayor interés es la computación inspirada en la naturaleza puesto que esta busca desarrollar técnicas de solución de problemas teniendo como inspiración a la naturaleza, enfocándose en la solución a problemas de alta complejidad. Por esto mismo, el interés hacia este campo ha aumentado, siendo cada vez más las personas interesadas en este campo que permite solucionar problemas en diversas áreas de la ingeniería.

El problema al que se da solución mediante este proyecto es la falta de profesionales formados en esta área; la razón es que hay una carencia de cursos que estén dirigidos a estudiantes de pregrado donde el material educativo esté en español y presenten una profundidad y anchura temática adecuada. Resolver este problema es importante ya que se atraen más estudiantes al área donde estos obtendrán un conocimiento interdisciplinar útil para solucionar problemas siendo esto una habilidad muy valiosa actualmente. Por estas razones se decidió construir un curso de nivel introductorio en esta área, dicho curso está enfocado en la computación inspirada por la naturaleza ya que esta área es la más llamativa en la ingeniería y además se evidencia la necesidad de un material como este en español que además sea completo y fácil de entender.

El objetivo general de este proyecto es diseñar un curso que introduzca a los estudiantes en tres tendencias principales de esta área – computación neuronal, computación evolutiva y computación de enjambres – para que los estudiantes puedan comprender, proponer, desarrollar y evaluar soluciones a problemas que son inviables usando la computación tradicional. Por cada tendencia se realizó la creación o curaduría de los recursos de aprendizaje necesarios. Las actividades y los recursos de aprendizaje se despliegan en una aula virtual siguiendo la estructura propuesta del curso para las tres secciones o tendencias.

El estudiante al finalizar el curso (i) tendrá conocimientos sobre las bases biológicas e informáticas de las principales tendencias de computación inspirada en la naturaleza, (ii) comprenderá los comportamientos biológicos y las meta-heurísticas asociadas al algoritmo principal de cada tendencia, y (iii) estará en la capacidad de resolver problemas usando enfoques de computación bioinspirada que de otra manera serían inviables. Además el estudiante (iv) propondrá, comprenderá y evaluará soluciones inspiradas en la naturaleza. Finalmente, gracias a todo, esto (v) se fortalecen conocimientos interdisciplinarios como habilidades investigativas y el pensamiento crítico.

Entre los recursos creados se encuentra un capítulo de un texto guía el cual detalla el aspecto biológico de la tendencia en sí para luego continuar con el aspecto informático del algoritmo correspondiente, finalizando con la metodología de solución de problemas en el marco del algoritmo específico.

Por otra parte, a modo de apoyo para el texto guía, se seleccionó un vídeo teórico que explica los detalles biológicos y un simulador visual del algoritmo correspondiente los cuales tienen el objetivo de acercar al estudiante de forma visual hacia el funcionamiento del algoritmo. Tanto para los vídeos como los simuladores se realizó un proceso de curaduría con el fin de elegir los mejores recursos.

Siguiendo con un enfoque más práctico, se construyó un *framework* por cada tendencia que se abarca en el curso, esto con el objetivo de facilitar las herramientas necesarias para que el estudiante construya soluciones de forma fácil e intuitiva. Además, se diseñaron una serie de laboratorios prácticos donde se propone la implementación y la aplicación del algoritmo o modelo correspondiente para solucionar un problema, usando dicho *framework*.

Finalmente una vez el estudiante adquiere este conocimiento y lo pone en práctica este realizará la búsqueda, selección y evaluación de una serie de artículos de aplicación para finalizar con el desarrollo de un proyecto.

El diseño de aprendizaje de la asignatura y sus respectivos recursos fueron sometidos a un proceso de revisión externa donde dos personas estudiaron y analizaron el material propuesto. De esta revisión se obtuvieron comentarios y sugerencias así como los puntos positivos que resaltaron. La opinión sobre el material fue muy buena: los dos revisores están de acuerdo con que (i) gracias a los distintos recursos el estudiante logra adquirir los conocimientos de una forma muy intuitiva y completa, (ii) los recursos presentados son de calidad, resaltando el valor e interés que estos generan; y (ii) la asignatura cuenta con un alcance y profundidad adecuados.

Índice general

Introducción

Motivación

La computación natural es un campo de investigación que tiene como propósito comprender los procesos computacionales que tienen lugar en la naturaleza y desarrollar modelos computacionales inspirados en la naturaleza. Este campo se divide en tres áreas: (i) la computación inspirada por la naturaleza, que toma inspiración de la naturaleza para desarrollar técnicas de solución de problemas; (ii) la simulación de la naturaleza por medio de la computación, que busca simular fenómenos naturales para facilitar su estudio; y (iii) la computación con materiales naturales, que busca sustituir el paradigma de computadores basados en silicona por el uso de otros materiales naturales.

Para la ingeniería, el área de mayor interés es la computación inspirada en naturaleza puesto que ofrece soluciones alternativas a problemas de alta dificultad. Estas alternativas de solución se basan en imitar comportamientos inteligentes observados en fenómenos naturales. Estos comportamientos son dignos de imitar puesto que son soluciones a problemas presentes en la naturaleza que se han perfeccionado durante los años por medio de la evolución.

Este campo de investigación ha permitido dar solución a diferentes tipos de problemas difíciles: optimización, clasificación, predicción, reconocimiento de patrones, entre otros; y ha evidenciado su potencial para ser útil en varios sectores; entre estos es posible mencionar el diseño de acueductos, distribución de la carga de trabajo entre nodos de un balanceador de carga en la nube, optimización de consumo de energía y planificación de rutas.

Por todo lo anterior, el interés hacia este campo ha aumentado, siendo cada vez más las personas interesadas en este fascinante campo que permite solucionar problemas en diversas áreas de la ingeniería.

Dentro de la oferta académica disponible sobre computación natural podemos encontrar una variedad de cursos ofrecidos por universidades internacionales: desde cursos especializados de pregrado hasta maestrías en el área. Además, se han escrito varios libros, la mayoría en inglés, donde se presenta este tema de una forma profunda y rigurosa. Sin embargo, estos materiales no son adecuados para público no especializado y no hemos encontrado un curso o texto que integre de forma adecuada las diferentes tendencias de esta área.

En el programa de Ingeniería de Sistemas de la Escuela Colombiana de Ingeniería actualmente se ofrece la asignatura Inteligencia Artificial: Computación Natural (**IACN**) que hace parte del énfasis de inteligencia artificial con una exigencia académica de 4 créditos. En el desarrollo de esta asignatura adquirimos un conjunto de conceptos y metodologías que nos permitieron desarrollar varios proyectos exitosos en el área y nos motivaron a proponer este trabajo dirigido.

Al finalizar **IACN** encontramos que los conocimientos adquiridos pueden ser muy importantes para cualquier estudiante de ingeniería. Al conocer esta área se adquieren un conjunto de herramientas importantes para solucionar problemas difíciles presentes en distintas disciplinas; además, permite a los ingenieros de sistemas desarrollar un pensamiento interdisciplinario en la creación de soluciones de software.

Es por todo esto que nuestra propuesta de diseñar de un material de aprendizaje en español que cubra de forma general, introductoria y práctica el área de la computación inspirada en la naturaleza es un aporte verdadero e importante en el aprendizaje de la comunidad estudiantil. Por otra parte, contribuye a la formación de profesionales interdisciplinarios siendo esto fundamental en tiempos de constante cambio y disrupción.

Objetivos

El objetivo general del proyecto es proponer un diseño de aprendizaje y desarrollar material de aprendizaje enfocado en la computación natural específicamente en la computación inspirada por la naturaleza. Los objetivos específicos son:

- Elaborar el **diseño** de la propuesta **de aprendizaje**
- Desarrollar un **texto guía** que proporcione una introducción de los conceptos biológicos e informáticos necesarios para algunas tendencias de la computación inspirada por la naturaleza.
- Seleccionar o de ser necesario desarrollar **recursos audiovisuales** para apoyar el entendimiento del comportamiento biológico y el funcionamiento de los algoritmos de los distintos fenómenos naturales.
- Seleccionar o plantear un **banco de retos** (preguntas, ejercicios y problemas) de interés para cada una de las tendencias.
- Implementar un ***framework*** con los principales algoritmos de cada tendencia, para que el estudiante pueda experimentar con los algoritmos y plantear soluciones a los problemas a partir de un código base.
- Desarrollar un conjunto de **laboratorios** que sirvan al estudiante para aplicar los conceptos teóricos aprendidos

Se plantea hacer un diseño de aprendizaje general y desarrollar el material de aprendizaje para tres tendencias de la computación inspirada por la naturaleza incluyendo el algoritmo más representativo de cada una:

- Computación neuronal: Perceptrón de una capa
- Computación evolutiva: Algoritmos genéticos
- Inteligencia de enjambres: Algoritmo de optimización de hormigas

Metodología

Para el desarrollo de este proyecto tiene como primer paso la elaboración de un **diseño de aprendizaje** donde las actividades que lo componen hacen uso de los diferentes recursos que conforman el material de aprendizaje.

Para el desarrollo de cada uno de los recursos de aprendizaje se usará una metodología específica, estos componentes son:

- Texto guía. El desarrollo del texto guía se inicia con la revisión de marco teórico general para definir la estructura temática que tendrá el libro, donde cada sección corresponde a un tendencia de la computación inspirada por la naturaleza. A partir de aquí, se empezará con la fase de escritura del texto guía donde por cada sección se realizará la revisión de marco teórico correspondiente. El objetivo de este componente es brindar la teoría biológica e informática necesaria para tener un aprendizaje concreto sobre cada uno de los temas.
- Recursos audiovisuales. Se realizará la respectiva curaduría, y en caso de ser necesario el desarrollo, de recursos audiovisuales que sirvan de apoyo visual para el entendimiento comportamiento biológico (videos) y la comprensión del funcionamiento de los distintos algoritmos (simuladores).

- Banco de retos. Se seleccionarán, adaptarán y propondrán preguntas, ejercicios y problemas significativos que permitan evidenciar las fortalezas y límites de las diferentes meta-heurísticas presentes en las tendencias.
- *Framework*. Se desarrollará un *framework* cuyo objetivo es ofrecer la implementación de los distintos algoritmos en un solo producto de software robusto, de tal manera que este producto apoye la elaboración de los laboratorios. Para el desarrollo de este se usará la metodología ágil de software, empezando por el diseño conceptual para luego implementar tres ciclos de desarrollo uno asociado a cada tendencia. El *framework* se validará con la solución de un problema paradigmático de la tendencia.
- Laboratorios. Se diseñará un conjunto de laboratorios donde cada uno corresponde a una tendencia. El objetivo de estos es poner en práctica los conocimientos adquiridos mediante la teoría presente en el texto en dos ámbitos: implementación de la meta-heurística y aplicación de la misma en la solución de problemas. Una vez terminados estos laboratorios el estudiante contará con un producto de software capaz de dar solución a un problema real y con la solución de un problema de interés en el área. Dichos laboratorios serán validados de tal manera que tengan la complejidad adecuada de acuerdo al alcance del curso.

Plan de trabajo

Para construir este material se realizarán las siguientes actividades

1. Desarrollo del diseño de aprendizaje
2. Desarrollo del material de aprendizaje correspondiente para cada una de las tendencias seleccionadas.
   - 2.1 Escritura en el texto guía
   - 2.2 Selección y/o implementación de simuladores
   - 2.3 Selección, adaptación y creación de problemas significativos
   - 2.4 Diseño, construcción y prueba de los algoritmos en el *framework*
   - 2.5 Diseño, desarrollo y validación de los laboratorios
   - 2.6 Revisión, validación e iteración del material desarrollado
3. Despliegue del curso
4. Evaluación del curso

Adicionalmente, paralelo a la creación del material de aprendizaje, se adelantarán actividades para conocer los avances de trabajos relacionados y desarrollar los entregables del trabajo dirigido:

0. Revisión de estado de arte
6. Escritura del artículo
7. Escritura del libro de proyecto
8. Socialización de resultados

Contenido

El documento está estructurado en las siguientes secciones.

	Actividad	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
	0	X	X	X			X	X				X	X					
	1	X	X				X					X				X		
Neuronal	2.1 y 2.3		X	X														
	2.2 y 2.3			X														
	2.4				X	X												
	2.5					X												
Evolutiva	2.1 y 2.3						X	X										
	2.2 y 2.3							X										
	2.4								X	X								
	2.5									X								
Enjambres	2.1 y 2.3										X	X						
	2.2 y 2.3											X						
	2.4												X	X				
	2.5													X				
	2.6					X				X				X	X	X	X	
	3														X	X	X	
	4															X	X	X
	6					X				X						X	X	
	7						X					X			X	X	X	
	8																	X
Avance							X					X						X
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17

Cuadro 1: Cronograma

- Introducción
- Capitulo I. Artículos
- Capitulo II. Texto guía
- Capitulo III. Prácticas
- Conclusiones y trabajo futuro

Parte I

Artículos

Capítulo 1

Artículo de revisión

1.1. Introducción

La computación natural es un campo de investigación que tiene como propósito comprender los procesos computacionales que tienen lugar en la naturaleza y desarrollar modelos computacionales inspirados en la naturaleza. Este campo se divide en tres áreas: (i) la computación inspirada por la naturaleza, que toma inspiración de la naturaleza para desarrollar técnicas de solución de problemas; (ii) la simulación de la naturaleza por medio de la computación, que busca simular fenómenos naturales para facilitar su estudio y (iii) la computación con materiales naturales, que busca sustituir el paradigma de computadores basados en silicona por el uso de otros materiales naturales[21].

Para la ingeniería, la tendencia de mayor interés es la computación inspirada en naturaleza puesto que ofrece soluciones alternativas a problemas de alta dificultad. Estas alternativas de solución se basan en imitar comportamientos inteligentes observados en fenómenos naturales[21]. Estos comportamientos son dignos de imitar puesto que son soluciones a problemas presentes en la naturaleza que se han perfeccionado durante los años por medio de la evolución[13].

Estas soluciones han mostrado efectividad en varios sectores de la ingeniería, como por ejemplo el diseño de acueductos [50], distribución de la carga entre nodos de un balanceador de carga en la nube [64], optimización de consumo de energía [22], planificación de rutas [83],entre otros. Por lo anterior, el interés hacia este campo ha aumentado, siendo cada vez más las personas interesadas en este fascinante campo que permite solucionar problemas en diversas áreas de la ingeniería.[23].

La pregunta de investigación de esta revisión es ¿Cómo se está enseñando la computación inspirada por la naturaleza?. El propósito de responder esta pregunta es conocer los componentes de los diseños de aprendizaje —el perfil, los recursos de aprendizaje, los prerrequisitos, las metodologías y actividades — utilizados en la enseñanza de esta área para en un trabajo futuro proponer un diseño de aprendizaje. Alrededor de esta pregunta se realizó una revisión tanto de artículos, como de currículos de cursos ofrecidos por universidades o presentes en la web.

Esta revisión se compone de dos partes. En la primera se analizan artículos de investigación relacionados a metodologías,experiencias y herramientas de aprendizaje utilizadas en la enseñanza de la computación bioinspirada en tres de sus principales tendencias: inteligencia de enjambre, computación evolutiva, computación neuronal y sistemas inmunes artificiales. En la segunda parte se analizan cursos de computación bioinspirada, o de alguna de sus ramas, encontrados tanto en páginas de universidades como en distintos portales web.

El objetivo de este trabajo es establecer una base teórica que responda a la siguiente pregunta: ¿Cómo se está enseñando computación bioinspirada?. El propósito de responder esta pregunta es conocer los elementos de los diseños de aprendizaje comunes relacionados a componentes como el perfil, prerrequisitos, las actividades y los recursos de aprendizaje utilizados en la enseñanza de esta área. Estos elementos serán la base para en un trabajo futuro construir un esquema base de un diseño de aprendizaje para un curso en el área.

1.2. Metodología

A continuación se describe la metodología definida para realizar este trabajo de revisión:

1. Definición de las preguntas objetivo
2. Revisión de artículos
    2.1 Definición de repositorios
    2.2 Definición de criterios de búsqueda
    2.3 Definición de criterios de selección
    2.4 Evaluación y selección de artículos
3. Revisión de cursos
    3.1 Definición de repositorios
    3.2 Definición de criterios de búsqueda
    3.3 Definición de criterios de selección
    3.4 Evaluación y selección de material
4. Desarrollo de las preguntas objetivo

1.2.1. Definición de preguntas objetivo

El objetivo de este trabajo de revisión es responder a la pregunta ¿Cómo se está enseñando computación bioinspirada?. Para responder con la pregunta objetivo se plantearon una serie de preguntas específicas que nos servirán para entender los distintos componentes de diseño de aprendizaje presentes en el material encontrado. Los componentes a tener en cuenta son el perfil, los prerrequisitos, las actividades de aprendizaje, los recursos de aprendizaje, para cada uno de estos componentes se planteó una pregunta. Finalmente surgió la pregunta de ¿Cómo evalúan los estudiantes este tipo de cursos? y se responderá porque permite ampliar la perspectiva de la pregunta objetivo para además poder conocer que piensan los estudiantes de los componentes que se están utilizando.

1. ¿Qué perfil tienen los estudiantes?
2. ¿Qué prerrequisitos tienen los cursos?
3. ¿Qué actividades de aprendizaje se utilizan?
4. ¿Qué recursos de aprendizaje se utilizan?
5. ¿Cómo evalúan los estudiantes este tipo de cursos?

Esperamos poder extraer un conjunto de metodologías, tendencias y experiencias comunes relacionadas a la enseñanza de la computación bioinspirada y sus tendencias.

1.2.2. Revisión de artículos

Definición de repositorios

Los repositorios seleccionados para realizar la búsqueda son cuatro motores de búsqueda académicos: *IEEE Xplore*, *Semantic Scholar*, *Google Scholar* y *SciELO*. Se considera que estos repositorios nos ofrecen la amplitud suficiente para realizar una buena búsqueda, puesto que plataformas como *Semantic Scholar* y *Google Scholar* permiten encontrar material en muchas otras plataformas.

Definición de criterios de búsqueda

Como criterios de búsqueda, se definieron expresiones regulares como *"[teaching |learning] {replacement}"*, *"[teaching |learning] tool for {replacement}"*, *"{replacement} course"* y *"{replacement} simulator"*, el campo *"{replacement}"* fue reemplazado por los siguientes valores que corresponden el área de computación natural y sus subramas:

- *Natural [Computing |Algorithms |Systems]*
- *Genetic [Computing |Algorithms |Systems]*
- *Swarm Intelligence*
- *Evolutionary [Computing |Algorithms |Systems]*
- *Neural Networks*
- *Artificial Inmune [Computing |Algorithms |Systems]*

En el caso del repositorio *SciELO* se utilizaron los equivalentes en español a los términos definidos.

Adicionalmente, se extendió la búsqueda sobre las referencias de los artículos encontrados, con el objetivo de encontrar material adicional útil para la revisión que pudiera haber sido excluido en los términos de búsqueda iniciales.

Definición de criterios de evaluación

Para la selección de artículos, se analizaron las secciones de resumen y conclusiones de los artículos encontrados. En los artículos presentaban herramientas de aprendizaje, se revisó adicionalmente que incluyeran contenido relacionado a las experiencias de aprendizaje generadas con la herramienta y no solo detalles de implementación. A partir de lo encontrado en esta búsqueda se aplicó un conjunto de criterios de inclusión que se presentan a continuación, para que un artículo sea seleccionado debe cumplir con al menos uno de estos criterios:

- Criterios de inclusión
    1. Se refiere a experiencias de profesores dictando cursos en las áreas relacionadas a computación inspirada por la naturaleza
    2. Se refiere a herramientas utilizadas para la enseñanza de computación inspirada por la naturaleza o alguna de sus subramas, explicando no solo las funciones de la herramienta sino su uso pedagógico y sus efectos en la enseñanza.
    3. Expone una perspectiva general del proceso de enseñanza del área de computación natural

Posterior a esta selección se utilizarán las siguientes métricas para evaluar los artículos.

- Interés
- Pertinencia
- Claridad
- Completitud
- Año
- Reconocimiento

Evaluación y selección de artículos

Teniendo en cuenta el contenido de los artículos se decidió priorizar ciertos aspectos en la lectura como se detalla a continuación:

- En los artículos relacionados a herramientas de aprendizaje, se priorizan las partes que detallan como se utilizan estas herramientas en la enseñanza de computación natural, por encima de los detalles de implementación .
- En los artículos que presenten contenido mixto entre temas de computación basada por la naturaleza y otras tendencias, solo se tendrán en cuenta las partes relacionadas a temas de computación basada por la naturaleza.
- Los demás artículos fueron leídos en su totalidad

En la tabla 1.1 se evaluaron lo artículos preseleccionados utilizando las métricas definidas anteriormente, adicionalmente se menciona que criterios de selección cumple el artículo.

Título	Interés	Pertinencia	Claridad	Completitud	Año	Reconocimiento	Criterios
Teaching Intelligent Systems at the University of Cincinnati	5	3,5	4	4	2011	0	1
Experience teaching a graduate level course in evolutionary computation	5	4	4	4	2009	2	1
A Virtual Laboratory on Natural Computing: A Learning Experiment	5	4	4	5	2008	5	1,2
Evolutionary Computation Teaching at Birmingham	3	3	3	3	1999	5	1,3
GraphEA: A 3D Educational Tool for Genetic Algorithms	5	4	4	4	2013	3	2
Experiences With Teaching Adaptive Optimization to Engineering Graduate Students	5	4	4	4	1999	3	1
A 'Hands on' Strategy for Teaching Genetic Algorithms to Undergraduates	3	3.5	4	4	2007	3	2
Teaching Evolutionary Algorithms	5	4.5	4	4.5	1999	1	1
Teaching .Evolutionary Computation" at the University of Stuttgart	5	4	4	4.5	2001	0	1
An educational tool for the ant colony optimization algorithm	5	4.5	4	4	2009	7	2
Teaching natural computation	5	4	3.5	4	2009	2	3
A Survey of Current Practice and Teaching of AI	4	3.5	4	3.5	2016	40	3
A web-based tool for teaching neural network concepts	5	3.5	4	4	2010	19	2

Cuadro 1.1: Evaluación de artículos

1.2.3. Revisión de cursos

Definición de criterios de búsqueda

Como criterios de búsqueda, se definieron términos base como *"{replacement} course syllabus"*, *"syllabus for {replacement}"* el campo *"{replacement}"* fue reemplazado por los siguientes valores que corresponden el área de computación natural y sus subramas:

- *Natural Computing*
- *Natural Computation*
- *Nature-Inspired Computing*

También se utilizaron los equivalentes en español a estos términos para ampliar la búsqueda a cursos en este idioma.

Definición de criterios de selección

Los cursos a tener en cuenta deben cumplir con alguno de los siguientes criterios de selección:

- El contenido publicado debe mostrar que el curso contiene contenido de al menos tres de las tendencias de la computación inspirada por la naturaleza.
- Para los cursos de universidad, debe ser clara la metodología utilizada en el curso

Selección de material

Los cursos encontrados fueron los siguientes:

1. Undergraduate Course: Natural Computing (INFR11161) - THE UNIVERSITY of EDINBURGH (2018) [65]
2. IE 590: Nature-Inspired Computing - Purdue University (2016) [81]
3. Cursos: Inteligencia Artificial - Fernando Sancho Caparrini[19]

1.3. ¿Cómo se está enseñando computación inspirada por la naturaleza?

El área de la computación natural se ha estado enseñando desde los comienzos de los 90[68] y desde entonces ha tenido una tendencia creciente tanto en el número de universidades que lo enseñan[65][81], como en la cantidad de material disponible en línea.

1.3.1. ¿Qué perfil de estudiantes se maneja ?

El perfil de la mayoría de los cursos analizados corresponde a (i) estudiantes de postgrado o estudiantes de pregrado de semestres avanzados de carreras de matemáticas o de ingeniería de computación, eléctrica, industrial o mecánica; (ii) con familiaridad con algún lenguaje de programación; e (iii) interesados en el aprendizaje de técnicas de solución de problemas alternativos a los presentes en la computación clásica [78][79][68][84]. En algunos casos se menciona como deseable (iv) el conocimiento básico en áreas como el cálculo, el álgebra y la probabilidad [23] [49]

También se encontraron algunos cursos dirigidos directamente a estudiantes de pregrado [27][65][82] y un curso menciona como otro posible perfil el de profesionales en ejercicio que deseen mejorar su experiencia de aprendizaje de posgrado mediante técnicas y herramientas que lo ayuden a encontrar mejores soluciones a problemas del mundo real[79].

1.3.2. ¿Qué prerrequisitos tienen los cursos?

El requerimiento común en los cursos encontrados, de forma explícita [23][79][78] o implícita por el tipo de actividades que se proponen [82] [81][19] [84][60], es el conocimiento en al menos un lenguaje de programación. Adicionalmente, en el curso de la universidad de Edimburgo se encontró como prerrequisito la formación en áreas como el álgebra lineal,cálculo y estadística [65]

1.3.3. ¿Qué actividades de aprendizaje se utilizan?

Se encontraron gran variedad de actividades en los cursos analizados, a continuación se mencionan algunos de los elementos que se mostraron con más frecuencia: (i) estudio de la teoría biológica; (ii) estudio de la teoría informática; (iii) tareas y ejercicios de programación correspondientes a la implementación del algoritmo correspondiente a la meta-heurística; (iv) aplicación del algoritmo en problemas interesantes para poder afianzar los conocimientos teóricos aprendidos por medio de la práctica.[60][27]; (v) la búsqueda, evaluación y presentación de artículos de investigación; y (vi) formulación e implementación de proyectos usando los algoritmos aprendidos.[23][79] [60] [84][49][78]

La primera actividad se justifica teniendo en cuenta que una de las principales dificultades que se ha encontrado al dictar estos cursos es que la mayoría de los estudiantes de ingeniería tienen muy poca o ninguna experiencia en áreas como la biología, la neurociencia o la química; por lo que la introducción a la inspiración biológica de cada algoritmo es fundamental. Otra consideración que se tiene en cuenta en los cursos es la necesidad de equilibrar el contenido teórico con actividades prácticas de solución de problemas incluyendo la teoría en las dos primeras actividades (i) y (ii);y la implementación y el uso de los algoritmos en problemas reales, en las actividades (ii), (iii) y (vi). Este balance es importante por dos factores : (i) son cursos que pueden ser tomados tanto por estudiantes avanzados de pregrado como de postgrado y (ii) la aplicación es un aspecto importante para la informática. Ciertos aspectos no pueden ser enseñados por medio de lecturas sino que deben experimentarse. [23] [60] [84] [49]

Adicionalmente se encontró el uso y la interacción con elementos didácticos o simuladores como una actividad menos común en los cursos pero de gran importancia para afianzar los conocimientos teóricos al incentivar la experimentación de los estudiantes sobre cada uno de los algoritmos sin la necesidad de implementarlos [19][82][27][55][41]. Otras actividades encontradas son la escritura de ensayos, la creación de *podcasts* explicando artículos y seminarios basados en artículos. [49]

1.3.4. ¿Qué recursos de aprendizaje se utilizan?

Los distintos recursos de aprendizaje encontrados se listan a continuación:

- Libros guía y artículos como fuente de información teórica [68][60][84][79][78][19]
- Simuladores virtuales [27][41][55][33][19]
- Ejercicios y problemas propuestos [27][78][82][68][79] [19]
- Paquetes de software que los estudiantes puedan extender en sus trabajos [49]
- Simuladores físicos(marionetas) [82]
- Banco de artículos para lectura y análisis [78]

1.3.5. ¿Cómo evalúan los estudiantes este tipo de cursos?

En general los estudiantes dan opiniones positivas sobre los cursos[78][23][84], algunos lo evaluaron como el más interesante e innovador en su carrera[49], otros comentaron que a través del curso, no sólo se han formado en estas técnicas, sino que, dada la naturaleza altamente práctica y experimental de este tipo de trabajos, tienen la oportunidad de desarrollar sus habilidades al solucionar problemas relacionados con las necesidades industriales o la investigación de vanguardia[79].

Finalmente, se encontraron dos observaciones con respecto al comportamiento de los estudiantes: (i) los estudiantes disfrutan conocer ideas que van más allá de los libros de texto de ingeniería, y aprecian especialmente la interdisciplinariedad del material [23]; y (ii) los estudiantes, en general, parecían más interesados en aquellos aspectos del curso que aplican estas ideas a la informática que los que tienen objetivos únicamente científicos. [49]

1.4. Conclusiones

Esta revisión presenta un conjunto de actividades, recursos y experiencias en común encontrados sobre como se esta enseñando computación inspirada por la naturaleza.

El perfil más común es el de estudiantes de postgrado; sin embargo, se encontraron algunos cursos recientes ofrecidos a estudiantes de pregrado. La inclusión de estudiantes de pregrado tiene sentido dado que el único prerrequisito es tener bases de programación, habilidad que se puede obtener en los primeros semestres; por otro lado, si bien es deseable tener algunos conocimientos matemáticos específicos, no es necesario que sea un requisito puesto que en el curso se puede enfocar sobre las ideas de cada propuesta y solo usar las matemáticas para darles rigurosidad[23][65][81]

Aunque no hay un diseño de aprendizaje formal y consensuado, hay varios aspectos en común como la enseñanza de la teoría biológica seguida de la teoría informática para luego enfocarse en la implementación y la solución de problemas; la implementación de proyectos que solucionan problemas prácticos y el estudio de artículos de investigación. Una posible justificación para este tipo de actividades es que ya que son cursos especializados que pueden ser tomados tanto por estudiantes de posgrado como de pregrado, es necesario equilibrar el contenido teórico riguroso con el material de utilidad inmediata.

El uso de simuladores como un recurso y las actividades relacionadas a la experimentación con los simuladores son elementos poco comunes en los cursos y consideramos que la inclusión de estos componentes es muy importante para llevar estos cursos a los estudiantes de pregrado, puesto que facilita la interacción con los algoritmos e incentiva la experimentación con estos sin necesidad de realizar algún tipo de implementación, por lo que el estudiante puede entender mejor el funcionamiento de cada algoritmo. Este tipo de recursos permiten una mejor comprensión tanto de la teoría como de la práctica, puesto que citando a Donald Knuth, "*An algorithm must be seen to be believed*"[51] y esto puede observarse en los resultados obtenidos por Decastro [27] donde los estudiantes opinaron que un curso con estas características es igual o mejor a un curso clásico.

Finalmente, el material en español encontrado es muy reducido, por lo que para nuevos estudiantes que no cuenten con un buen entendimiento del inglés, el lenguaje podría complicar un poco su aprendizaje. Aunque se encontró un curso en español bastante completo a nivel teórico y que utiliza simuladores para la explicación de los algoritmos, este no cuenta con un

conjunto de laboratorios que puedan servir al estudiante para afianzar sus conocimientos por medio de la solución de problemas.

1.5. Trabajo futuro

Teniendo en cuenta la poca presencia de cursos de nivel introductorio, de amplio cubrimiento y en español de computación inspirada por la naturaleza se propone la creación de un curso que tenga en cuenta los componentes de los diseños de aprendizaje exitosos encontrados en la literatura, que sea dirigido a estudiantes de semestres iniciales de pregrado y que sus recursos estén en español. Un punto inicial importante es el curso en español encontrado puesto que servirá como base para proponer puntos de mejora y poder desarrollar un material de nivel introductorio que permita al estudiante tener una buena experiencia de aprendizaje.

Capítulo 2

Diseño de aprendizaje

2.1. Introducción

La computación natural es un campo de investigación que tiene como propósito comprender los procesos computacionales que tienen lugar en la naturaleza y desarrollar modelos computacionales inspirados en la naturaleza. Este campo se divide en tres áreas: (i) la computación inspirada por la naturaleza, que toma inspiración de la naturaleza para desarrollar técnicas de solución de problemas; (ii) la simulación de la naturaleza por medio de la computación, que busca simular fenómenos naturales para facilitar su estudio y (iii) la computación con materiales naturales, que busca sustituir el paradigma de computadores basados en silicona por el uso de otros materiales naturales[21].

Para la ingeniería, la tendencia de mayor interés es la computación inspirada en naturaleza puesto que ofrece soluciones alternativas a problemas de alta dificultad. Estas alternativas de solución se basan en imitar comportamientos inteligentes observados en fenómenos naturales[21]. Estos comportamientos son dignos de imitar puesto que son soluciones a problemas presentes en la naturaleza que se han perfeccionado durante los años por medio de la evolución[13].

Estas soluciones han mostrado efectividad en varios sectores de la ingeniería, como por ejemplo el diseño de acueductos [50], distribución de la carga entre nodos de un balanceador de carga en la nube [64], optimización de consumo de energía [22], planificación de rutas [83],entre otros. Por lo anterior, el interés hacia este campo ha aumentado, siendo cada vez más las personas interesadas en este fascinante campo que permite solucionar problemas en diversas áreas de la ingeniería.[23].

Teniendo en cuenta el incremento de interés que ha tenido este campo, se decidió construir un curso en computación natural, enfocado a la computación inspirada por la naturaleza. Para esto, primero se realizó una revisión del estado del arte para determinar las características de los cursos dictados en los últimos años, tanto en universidades como presentes en la web. A grandes rasgos, se encontró que estos cursos suelen ser dictados principalmente a estudiantes de posgrado o estudiantes avanzados de pregrado, la mayoría está en inglés, no suelen incluir más de dos de las tendencias relacionadas a la computación inspirada por la naturaleza.

A partir de los resultados obtenidos en la investigación previa se evidencia la necesidad de un curso de computación natural en español de nivel introductorio ofrecido a estudiantes de semestres iniciales de pregrado. El objetivo de este artículo es presentar el diseño de aprendizaje que dirigirá el desarrollo del curso, para esto se tendrá en cuenta la investigación realizada, no solo para incluir diferenciadores, sino también para incluir estructuras de enseñanza que se han utilizado anteriormente y han dado resultados.

2.2. Identificación

2.2.1. Objetivos

El objetivo general de este curso es introducir a los estudiantes en el área de computación inspirada por la naturaleza; de modo que puedan comprender, proponer, desarrollar y evaluar soluciones a problemas difíciles que son inviables usando la computación tradicional.

Objetivos de aprendizaje El estudiante al finalizar el curso debe ser capaz de:

1. Conocer las bases biológicas e informáticas de las principales tendencias de computación inspirada en la naturaleza
2. Comprender los comportamientos biológicos y las meta-heurísticas asociadas a los algoritmos más reconocidos de las diferentes tendencias
3. Identificar los problemas que no tienen soluciones viables usando la computación tradicional y que podrían ser abordados con alguna estrategia de computación natural.
4. Diseñar e implementar soluciones informáticas utilizando los enfoques de la computación inspirada por la naturaleza
5. Proponer, comprender y evaluar soluciones inspiradas en la naturaleza

Competencias transversales Los estudiantes desarrollaran competencias para:

1. Utilizar los artículos técnicos como medio de transmisión de conocimiento
2. Realizar búsqueda, elección y evaluación crítica de artículos de aplicación
3. Desarrollar, presentar y analizar una solución siguiendo la metodología disciplinar

2.2.2. Perfil y contexto de uso

Estudiantes de pregrado de ingeniería, en el rango de edad de 17 a 23 años, que estén interesados en aprender a solucionar problemas construyendo soluciones de software inspiradas por los comportamientos inteligentes demostrados por la naturaleza y en desarrollar habilidades de experimentación e investigación. Los estudiantes deben tener bases de programación en el lenguaje Python y conocimientos de matemática básica. Es deseable el conocimiento en álgebra lineal y calculo diferencial pero no es un requisito del curso.

2.2.3. Lineamientos

Profundidad y Anchura La anchura se refiere a la cantidad de temas distintos tratados, mientras la profundidad se refiere al detalle presentado en cada uno de los temas. Considerando la anchura, en este curso se busca tener por anchura la presentación de al menos tres de las principales tendencias y por profundidad lo correspondiente a uno de los algoritmos más significativos de cada tendencia. Lo anterior para dar al estudiante una visión general de las áreas que componen la computación inspirada por la naturaleza pero también darle conocimiento profundo que le permita usar con propiedad un algoritmo de cada una de las tendencias.

Estructura de recursos que soporte y facilite el aprendizaje El uso de recursos para guiar el aprendizaje es de especial importancia en este curso principalmente por tres razones: (i) es esencial presentar conceptos biológicos con los que los estudiantes posiblemente no estén familiarizados, (ii) es importante que los estudiantes logren implementar los algoritmos seleccionados y (iii) es necesario que los estudiantes aprendan a solucionar problemas usando esas meta-heurísticas.

Los recursos ofrecidos en el curso deben guiar al estudiante en el proceso de obtención y validación de conocimientos y además debe presentar retos para que los conocimientos obtenidos puedan ponerse a prueba. El soporte ofrecido al estudiante se irá retirando a medida que se avanza en el aprendizaje.

La inclusión de retos es necesaria en el curso porque muchos de los conocimientos que se quieren entregar son solo obtenibles por medio de la práctica y la experimentación, por lo que el estudiante debe usar estos problemas para explorar, entender y apropiar el uso de cada algoritmo.

Brindar fundamentos teóricos, prácticos y metodológicos a nivel interdisciplinar El curso tiene como objetivo construir las bases de conocimiento necesarias para que los estudiantes puedan: (i) comprender los conceptos biológicos detrás de cada una de las tendencias, (ii) comprender los conceptos informáticos que describen los principales algoritmos de cada tendencia (en su versión base) (iii) implementar la versión base de cada algoritmo (iv) solucionar problemas utilizando los algoritmos implementados. Con esto se busca que el estudiante tenga un conocimiento general de cada una de las tendencias y cuenten con las herramientas necesarias para que, de ser necesario, puedan investigar e implementar versiones modificadas de los algoritmos.

Incentivar el trabajo en equipo Las actividades propuestas a los estudiantes deben tener la complejidad y extensión necesarios para permitir el trabajo en equipo, con esto se busca que los estudiantes se puedan apoyar en sus compañeros para solucionar dudas, analizar problemas e intercambiar conocimiento.

Brindar retrospectiva para afianzar conocimientos El curso debe presentar espacios de retrospectiva en los que se analicen las actividades realizadas y se extraigan lecciones de lo sucedido en cada una de ellas.

Proporcionar conocimiento para solucionar problemas Es de gran importancia que se presente una metodología de solución de problemas para que el estudiante entienda como puede solucionar problemas de distintos tipos usando herramientas presentadas en el curso. También es necesario presentar problemas desafiantes para el estudiante que lo lleven a poner en práctica lo aprendido a nivel teórico y práctico; y a desarrollar su habilidad de identificar y solucionar problemas utilizando métodos de computación inspirada por la naturaleza

Desarrollo de pensamiento crítico Con este curso se busca adicionalmente que los estudiantes desarrollen sus habilidades de análisis, evaluación y presentación de soluciones, con el objetivo que aprendan a obtener conocimiento a partir de las soluciones de otros, no solo a nivel de las tendencias presentadas en el curso, sino también como una herramienta más en su carrera profesional.

2.2.4. Diferenciadores

Considerando los cursos revisando en la investigación, este curso contará con los siguientes diferenciadores :

- El perfil del curso corresponde a estudiantes de pregrado de primeros semestres
- Los recursos y actividades propuestos permiten que el estudiante desarrolle sus conocimientos a nivel teórico y práctico.

- La presencia de recursos audiovisuales destinados a facilitar el entendimiento de cada uno de los temas tanto a nivel biológico como a nivel informático
- El curso presenta, a nivel introductorio, tres de los principales tendencias de la computación inspirada por la naturaleza , con la inclusión de el algoritmo más importantes de cada uno de las tendencias.
- El curso incluye actividades de búsqueda, selección y análisis de artículos de aplicación junto con actividades de desarrollo de problemas.
- El curso fomenta desarrollo de competencias transversales como la lectura crítica de literatura y la presentación de soluciones.

2.3. Diseño de Aprendizaje

2.3.1. Roles

1. Estudiante
2. Tutor

2.3.2. Actividades del estudiante

1. **Revisar fuentes de conocimiento (Por tema)**: El estudiante se apoyará en los recursos propuestos para obtener y desarrollar los conocimientos teóricos tanto biológicos como informáticos de cada uno de los temas. Con las actividades mencionadas a continuación se contribuye al cumplimiento de los objetivos de aprendizaje (1) y (2).
   1.1 Comprender teoría biológica. Leer texto guía y ver video.
   1.2 Comprender teoría informática. Leer texto guía y experimentar simulador.
   1.3 Comprender implementación de algoritmos. Revisar problema de ejemplo.
   1.4 Comprender metodología de solución. Leer texto guía y revisar problema ejemplo.
2. **Desarrollar los retos propuestos**: En este conjunto de actividades se plantean una serie de preguntas, problemas y ejercicios que buscan que el estudiante afiance los conocimientos teóricos obtenidos y desarrolle su habilidad de reconocer y solucionar problemas utilizando enfoques de computación inspirada por la naturaleza. Con las actividades mencionadas a continuación se contribuye al cumplimiento de los objetivos de aprendizaje (1),(2) y (4).
   2.1 Responder las preguntas
   2.2 Resolver ejercicios
   2.3 Solucionar problemas
3. **Desarrollar laboratorios (Por tema)**: El estudiante realizará la implementación del algoritmo base de cada uno de los temas, partiendo de una base de código que guía el desarrollo y la experimentación con un problema inicial, adicionalmente se plantea un problema desafiante de interés histórico y un conjunto de preguntas. Con las actividades mencionadas a continuación se contribuye al cumplimiento de los objetivos de aprendizaje (2),(3) y (4).
   3.1 Implementar zonas del algoritmo base
   3.2 Solucionar problema simple para experimentar implementación
   3.3 Solucionar problema utilizando el *Framework*

3.4 Desarrollar preguntas planteadas por el laboratorio

4. **Buscar, evaluar y presentar artículos de aplicación** El estudiante realizará la búsqueda y evaluación de un conjunto de artículos de aplicación que sean de su interés, de este conjunto seleccionará un artículo que analizará a profundidad y presentará a sus compañeros siguiendo la metodología disciplinar. Este conjunto de actividades busca contribuir al cumplimiento de los objetivos de aprendizaje (2) y (4), además de buscar que los estudiantes obtengan las competencias transversales (1) (2) y (3)
   4.1 Preseleccionar un grupo de artículos de aplicación (3 tentativamente). Teniendo en cuenta las preguntas planteadas en la metodología de presentación de soluciones
   4.2 Evaluar los artículos utilizando los criterios de evaluación
   4.3 Seleccionar el mejor artículo teniendo en cuenta la evaluación
   4.4 Analizar el artículo seleccionado
   4.5 Presentar a los compañeros el artículo teniendo en cuenta la metodología de presentación de soluciones
5. **Desarrollar un proyecto de curso** El estudiante seleccionará un problema de su interés y a partir de este realizará el análisis, diseño e implementación de una solución utilizando alguno de los enfoques vistos durante el curso. Este conjunto de actividades busca contribuir al cumplimiento de los objetivos de aprendizaje (3) y (4), además de buscar que los estudiantes obtengan la competencia transversal (3)
   5.1 Seleccionar un problema a resolver y explicar porqué es importante usar computación natural para resolverlo
   5.2 Analizar el problema
   5.3 Desarrollar la solución al problema teniendo en cuenta la metodología de presentación de soluciones

2.3.3. Actividades del tutor

1. Acompañar al estudiante en la revisión de fuentes de conocimiento y en el uso de los distintos recursos
2. Evaluar y retro-alimentar al estudiante en las distintas actividades del curso
   2.1 Retos
   2.2 Laboratorios
   2.3 Presentación de artículos
   2.4 Presentación de proyecto

2.3.4. Recursos

1. Texto guía
   1.1 Teoría biológica
   1.2 Teoría informática
   1.3 Metodología de solución de problemas
2. Recursos audiovisuales
   2.1 Videos
   2.2 Simuladores

3. Banco de retos
    3.1 Preguntas
    3.2 Ejercicios
    3.3 Problemas
4. Laboratorios
    4.1 Guía de los laboratorios
    4.2 Notebooks de Jupyter : esquema para implementación de zonas de algoritmo y problema simple de validación
    4.3 Notebooks de Jupyter : esquema para solución de problema
5. *Framework* de CNA
    5.1 API
    5.2 Problemas resueltos
6. Metodología de presentación de soluciones
7. Criterios de evaluación de artículos

2.3.5. Diagramas

A continuación se presentan los diagramas del diseño que ilustran las interacciones entre las actividades, los roles y los recursos propuestos.

Modelo pedagógico general

En la figura 2.1 se presenta el modelo pedagógico general que muestra, a nivel general, las actividades que realiza cada uno de los roles y los recursos que apoyan la realización de dichas actividades, adicionalmente, se seccionan las actividades en 3 zonas distintas: teoría,práctica e investigación

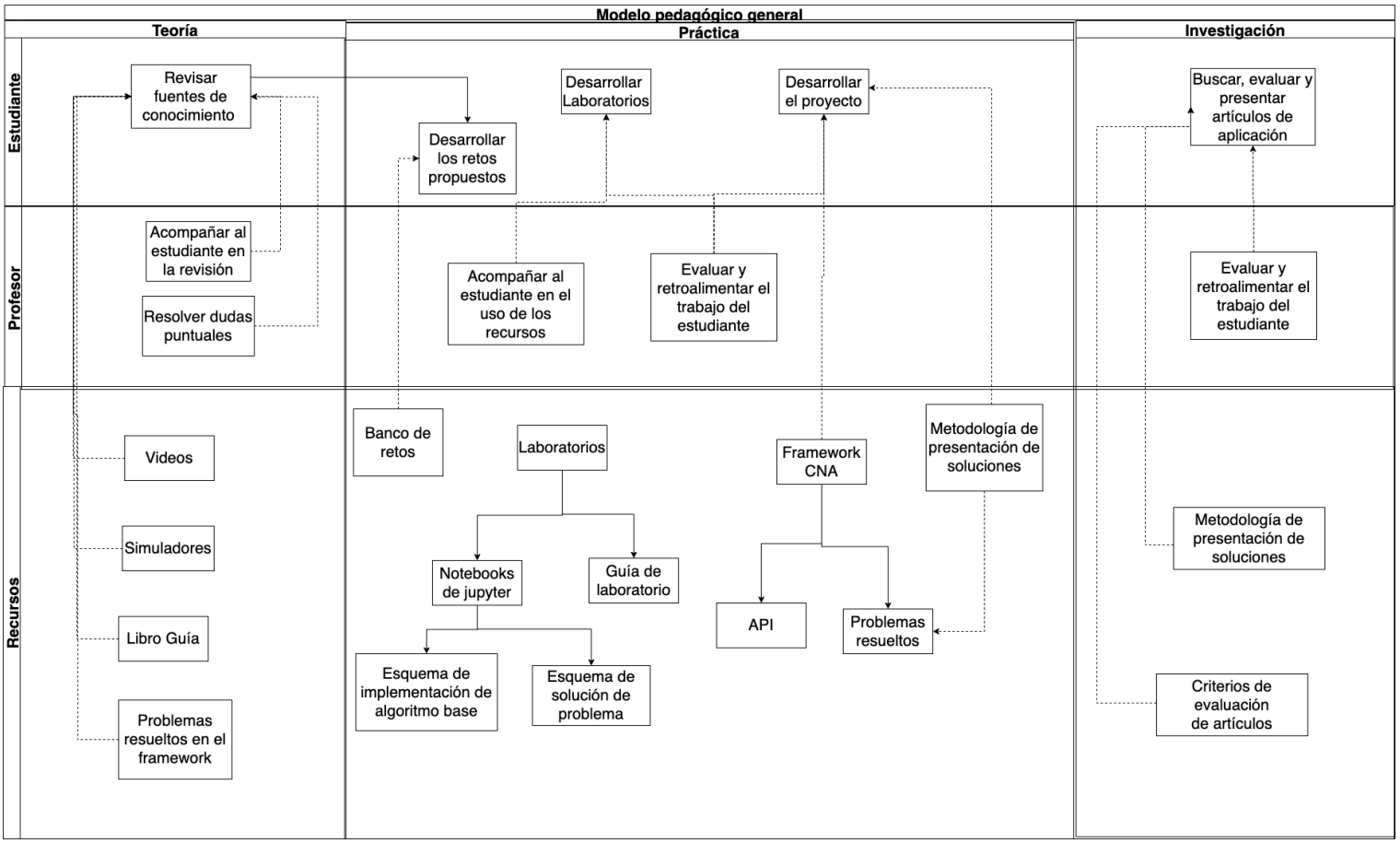


Figura 2.1: Modelo pedagógico general

Teoría

En la figura 2.2 se presenta con detalle las actividades y recursos presentes en la parte teórica del curso

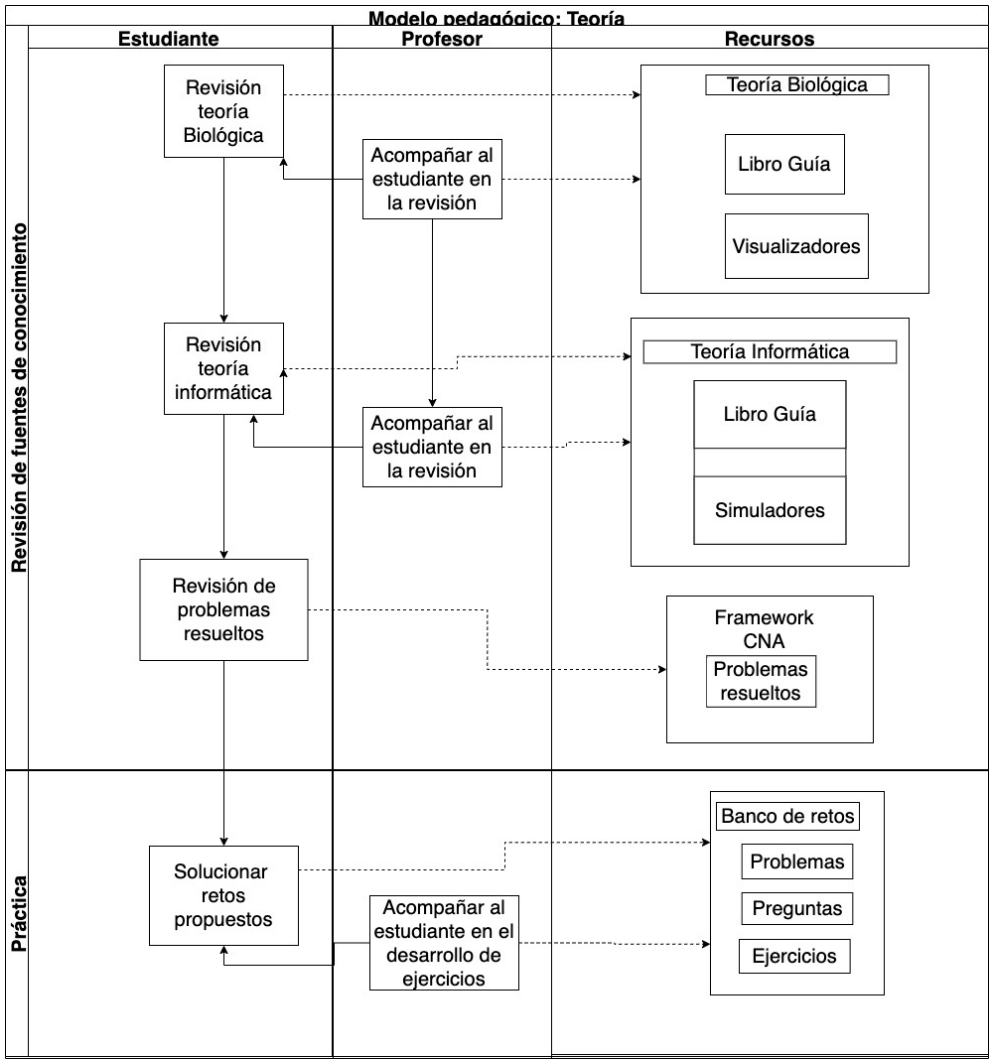


Figura 2.2: Modelo pedagógico: Teoría

Práctica - Laboratorios

En la figura 2.3 se presenta con detalle las actividades y recursos que se utilizan en la realización de los laboratorios del curso

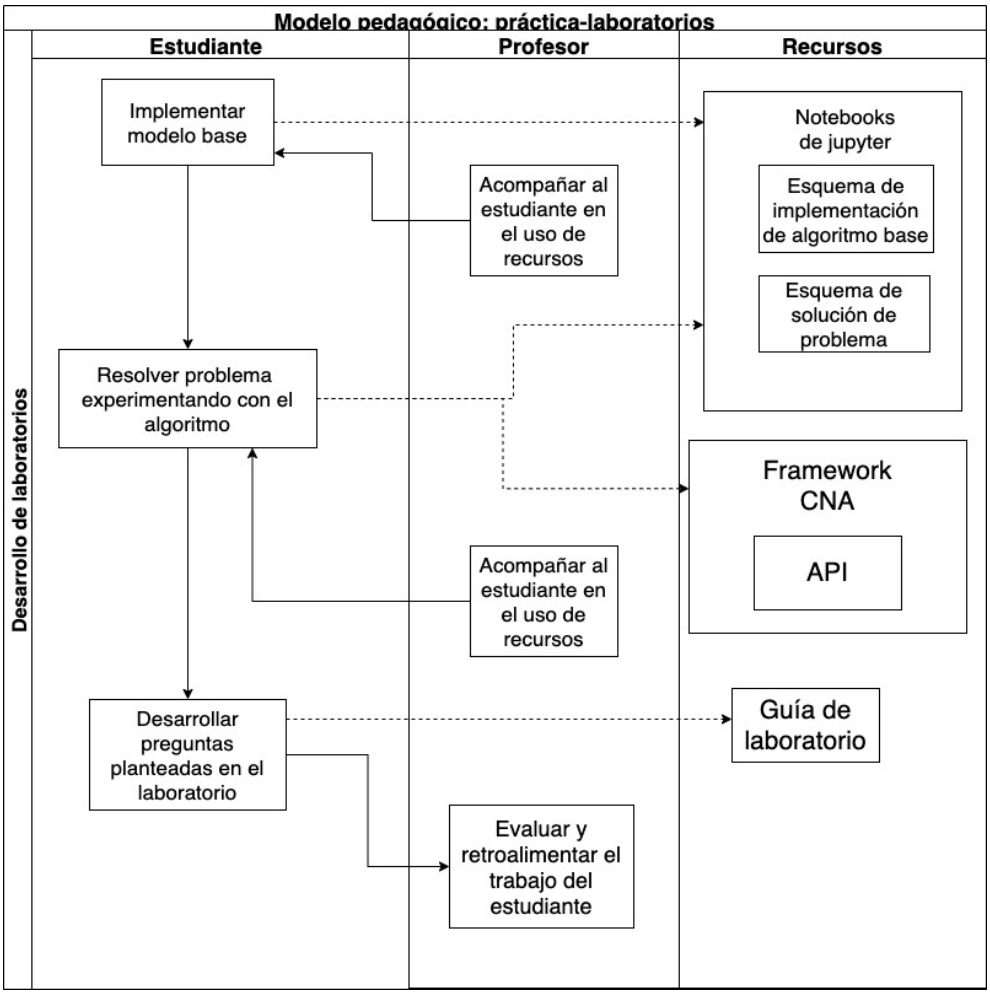


Figura 2.3: Modelo pedagógico: Práctica-Laboratorios

Práctica - Proyecto

En la figura 2.4 se presenta con detalle las actividades y recursos que se utilizan en la realización del proyecto del curso

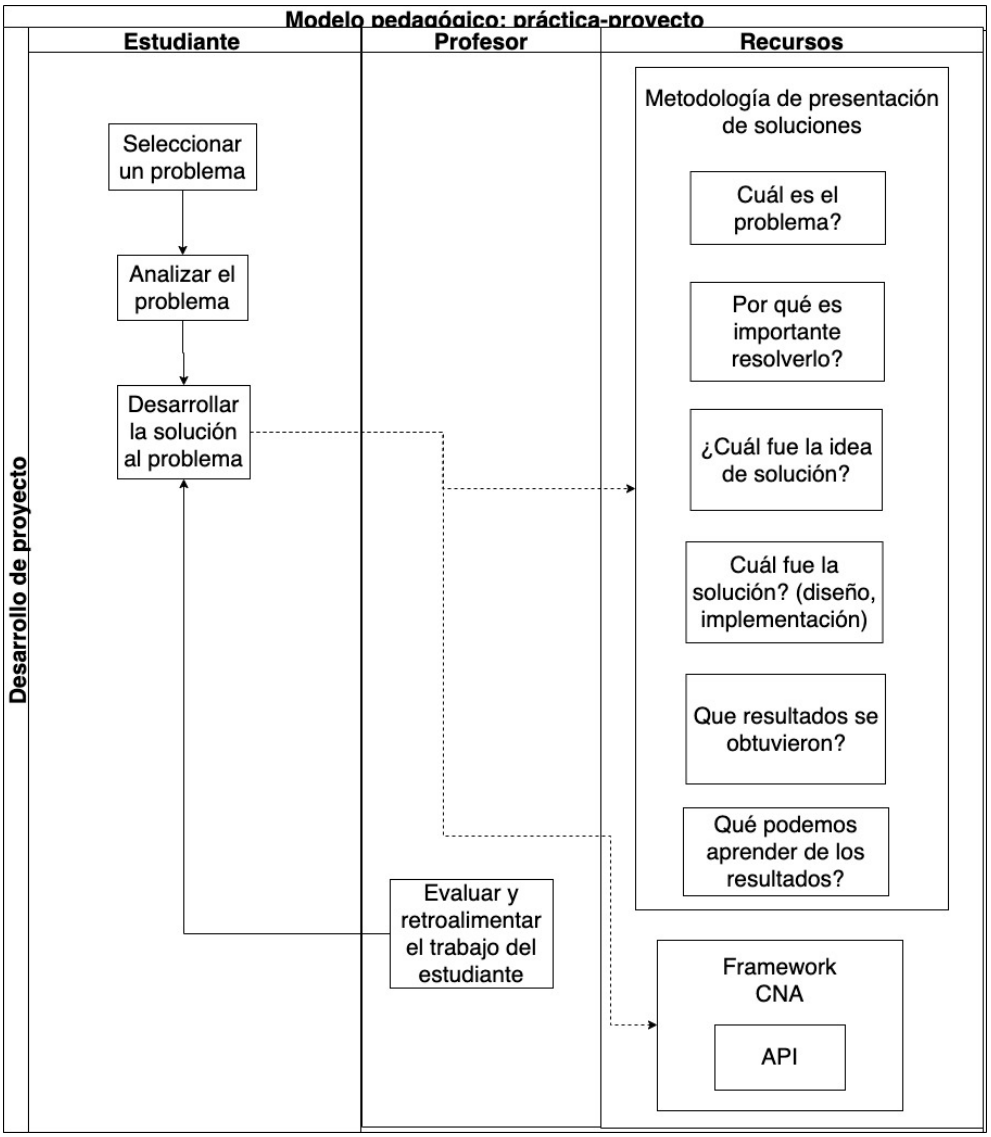


Figura 2.4: Modelo pedagógico: Práctica-Proyecto

Investigación

En la figura 2.5 se presenta con detalle las actividades y recursos que se utilizan en el componente de investigación del curso. Es importante destacar el uso de la metodología de presentación de soluciones para facilitar la búsqueda de artículos, puesto que los estudiantes no suelen tener experiencia en este tipo de actividades. El uso de esta metodología tiene como objetivo que los estudiantes puedan identificar rápidamente las ideas principales de las soluciones presentadas en los artículos para que las puedan tener en cuenta al momento de la evaluación de la solución y al momento de extraer los conocimientos presentados.

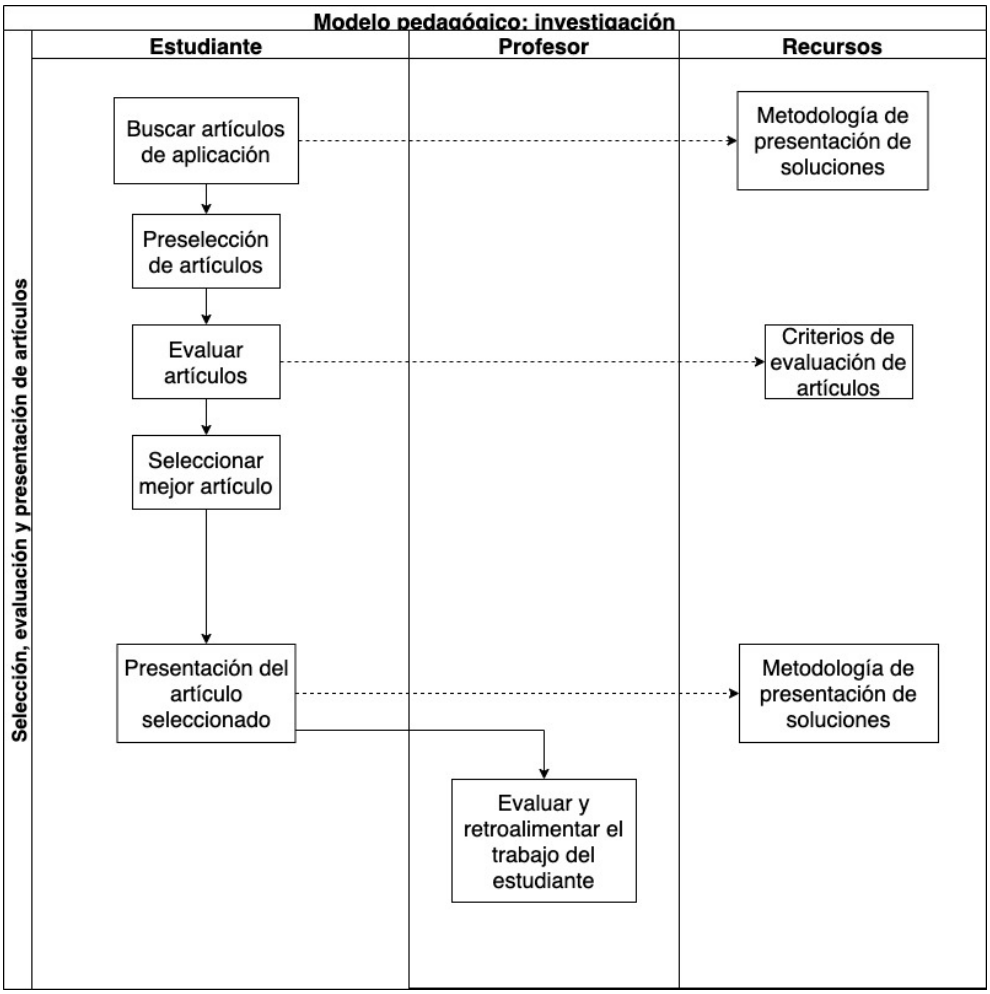


Figura 2.5: Modelo pedagógico: Investigación

Relaciones entre recursos

En la figura 2.6 se presenta un diagrama de clases que muestra las relaciones entre los recursos presentados, adicionalmente se muestra a que sección del curso pertenece cada recurso

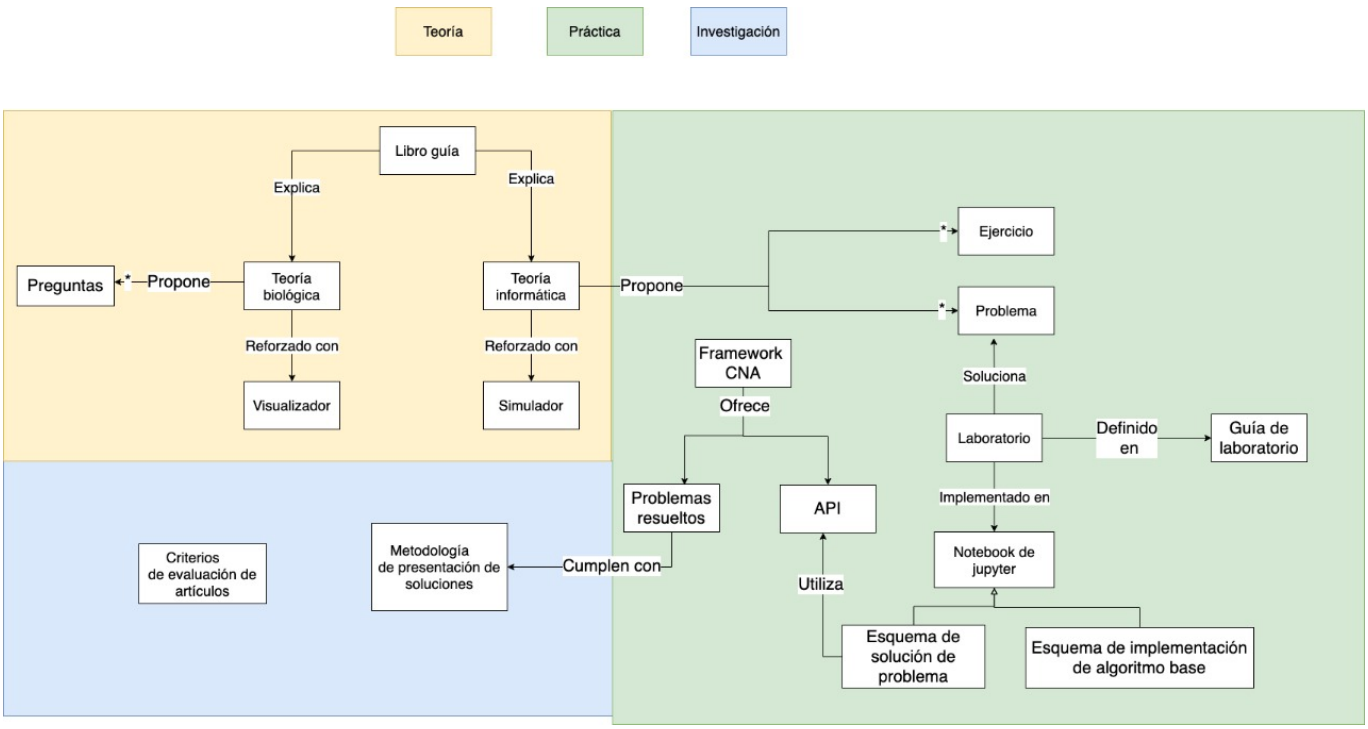


Figura 2.6: Modelo pedagógico: Diagrama de clases de recursos

2.3.6. Cronograma

A continuación se propone un cronograma para un curso de 17 semanas que incluye todas las actividades propuestas

Semana 1 Actividades en clase: Presentación del curso, introducción a computación natural y sus principales enfoques.
Trabajo autónomo previo a clase: lectura del capítulo de introducción

A partir de la semana 2 y hasta la semana 13 se repetirá la siguiente estructura de cuatro semanas para cada una de las tendencias. Primero se abordará la computación neuronal, seguido de la computación evolutiva y finalmente la computación basada en enjambres.

Clase teórica Actividades en clase: Apropiación de los conceptos teóricos, presentación de visualizadores y simuladores de cada tendencia.
Trabajo autónomo previo a clase: Lectura del capítulo correspondiente a cada tendencia.

Clase de laboratorios Actividades en clase: Inicio de implementación de los laboratorios.
Trabajo autónomo previo a clase: Experimentar con el simulador, buscar artículos de aplicación de interés de cada tendencia.

Clase de presentación de artículos Actividades en clase: Presentación de artículos de aplicación de redes neuronales.

Trabajo autónomo previo a clase: Desarrollo del laboratorio, construcción de presentación de artículo seleccionado.

Clase de cierre de tendencia Actividades en clase: Retroalimentación de trabajos realizados, espacio de retrospectiva, resumen de las lecciones aprendidas.

Semana 14 Actividades en clase: Socialización de problemas y diseños de solución. Trabajo autónomo previo a clase: Selección y análisis de problema a solucionar en el proyecto final, Escritura del artículo del proyecto.

Semana 15 Actividades en clase: Primera presentación del proyecto, Retroalimentación de las presentaciones Trabajo autónomo previo a clase: Implementación de la solución al problema solucionado, escritura del artículo del proyecto, construcción de presentación del proyecto.

Semana 16 Actividades en clase: Presentación final de proyectos Trabajo autónomo previo a clase: Implementación de correcciones a partir de los comentarios de la retroalimentación

Semana 17 Actividades en clase: Cierre del curso, espacio de retrospectiva y revisión de lo aprendido durante el semestre

2.3.7. Despliegue

Para desplegar los recursos planteados en el diseño de aprendizaje, se utilizó el ambiente virtual Moodle, se diseñó la siguiente estructura para presentar los recursos :

- Sección biológica
  - Lectura
  - Vídeos
  - Discusión
    - Preguntas de reflexión
- Sección informática
  - Lectura
  - Simulador
  - Discusión
    - Preguntas de reflexión
    - Ejercicio usando el simulador
  - *Framework*
    - Documentación del *Framework*
    - Problema resuelto usando el *Framework*
  - Laboratorio
    - Guía de laboratorio
    - Implementación algoritmo base
    - Problema propuesto

- Evaluación del módulo

Esta estructura se utilizó para presentar las 3 tendencias del curso, en la figura 2.7 podemos ver un ejemplo para la tendencia de computación neuronal

Figura 2.7: Despliegue de recursos:Computación neuronal

2.4. Recursos base

2.4.1. Metodología de presentación de soluciones

1. ¿Cuál es el problema?
2. ¿Por qué es importante resolverlo?
3. ¿Cuál fue la idea de solución?
4. ¿Cuál fue la solución? (diseño, implementación)?
5. ¿Qué resultados se obtuvieron?
6. ¿Qué podemos aprender de los resultados?

2.4.2. Criterios de evaluación de artículos

1. Interés: El problema resuelto en el artículo es de interés (1-5)
2. Pertinencia: En la solución se aplica la tecnología en estudio (1-5)
3. Claridad: El enfoque de solución es claro (1-5)
4. Completitud: Se presenta el detalle de la solución (1-5)
5. Efectividad: La solución tiene un buen desempeño (1-5)
6. Reconocimiento: Número de veces citado
7. Actualidad : Año de publicación

2.5. Evaluación

Con el objetivo de evaluar el diseño de aprendizaje y los recursos propuestos para cada tendencia, se realizó una versión piloto del curso con dos estudiantes, uno con conocimiento previo en el área y otro sin conocimiento previo, solo con conocimientos del lenguaje de programación python. Esta versión del curso se limitó a la realización de las siguientes actividades para cada una de las tendencias: (1) Revisar fuentes de conocimiento, (2) Desarrollar los retos propuestos y (3) Desarrollar laboratorios.

Para evaluar cuantitativamente esta versión del curso se utilizaron las métricas presentadas en la tabla 2.1, adicionalmente, se tuvieron en cuenta los comentarios y sugerencias de los estudiantes para evaluar el curso de forma cualitativa.

Criterio	Texto	Video	Simulador	Framework	Laboratorio: Algoritmo	Laboratorio: Problema
Claridad (1-5)	X					
Complejidad (Baja/ Adecuada/ Alta)					X	X
Facilidad de uso (1-5)			X	X		
Interés (1-5)		X			X	X
Extensión (Baja/ Adecuada/ Alta)					X	X
Utilidad (1-5)		X	X	X		

Cuadro 2.1: Criterios de evaluación de recursos de aprendizaje

A continuación se muestran los resultados cuantitativos y cualitativos obtenidos:

Evaluación cuantitativa En la tabla 2.2 se muestran los resultados a nivel cuantitativo de la evaluación realizada, para obtener estos valores se realizó un promedio de la evaluación de los estudiantes para cada una de las tendencias para los criterios de claridad, facilidad de uso, interés y utilidad. Para los criterios de extensión y complejidad se realizó la moda de los valores obtenidos, en los casos de empate se muestran todas las categorías involucradas.

Criterio	Texto	Video	Simulador	Framework	Laboratorio: Algoritmo	Laboratorio: Problema
Claridad (1-5)	4.1					
Complejidad (Baja/ Adecuada/ Alta)					Adecuada/ Alta	Alta
Facilidad de uso (1-5)			3.3	3.5		
Interés (1-5)		5			3.8	4
Extensión (Baja/ Adecuada/ Alta)					Adecuada	Adecuada
Utilidad (1-5)		5	3.8	3.5		

Cuadro 2.2: Evaluación cuantitativa de los recursos de aprendizaje

Evaluación cualitativa

- Estructura del curso
  - Fortalezas: Encontraron la estructura interesante, el diseño del curso permite una experiencia fluida y facilita el aprendizaje a partir de la interacción de las distintas actividades y recursos. Fue resaltado por los estudiantes como el mejor componente del curso.
  - Aspectos por mejorar y sugerencias: Presentaron la sugerencia de incluir pruebas automáticamente calificables (pruebas tipo test) para la validación de conocimientos
- Libro
  - Fortalezas: El contenido es interesante, permite comprender rápidamente los componentes más importantes de cada tendencia, claridad y buen manejo de imágenes en los capítulos de computación evolutiva e inteligencia de enjambres.
  - Aspectos por mejorar y sugerencias: Mejorar la redacción de algunas zonas, llevar las fortalezas de cada capítulo a los demás, incluir más imágenes y complementar las secciones de metodología
- Videos
  - Fortalezas: Permiten entender rápidamente los conceptos biológicos, son muy atractivos e interesantes, la duración es apropiada.
  - Aspectos por mejorar y sugerencias: No se recibieron comentarios relacionados a posibles mejoras o aspectos negativos
- Simuladores
  - Fortalezas: Útiles e interesantes para comprender el funcionamiento del algoritmo

    - Aspectos por mejorar y sugerencias: Cambiar el simulador de algoritmos genéticos porque es difícil de usar, mejorar los ejercicios que acompañan la experimentación

- *Framework*
    - Fortalezas: Los problemas resueltos interesantes y útiles para comprender la metodología utilizada al momento de solucionar problemas con cada una de las tendencias, el *Framework* es extensible y útil para solucionar los problemas.
    - Aspectos por mejorar y sugerencias: Mejorar la documentación del *Framework* para facilitar su uso y extensión, mejorar las zonas de análisis de resultados en los problemas resueltos, puesto que muchas veces no era claro como se llegó a los resultados mostrados.
- Laboratorios
    - Fortalezas: La experiencia guiada es agradable y facilita la realización de los laboratorios, es muy útil tener toda la información teórica a la mano para luego poder implementarla, los problemas propuestos son interesantes
    - Aspectos por mejorar y sugerencias:Diseñar pruebas que faciliten la búsqueda de errores, incluir recursos interactivos a los *notebooks*, mejorar la redacción y el flujo de algunos fragmentos

2.6. Conclusiones

En este trabajo se logró el diseño de aprendizaje de un curso de computación bioinspirada. Para esto se partió de las ideas encontradas en el estado del arte establecido en un trabajo previo y se realizó el diseño incluyendo distintas actividades y recursos para obtener una estructura sólida, con el objetivo de aportar buenas experiencias de aprendizaje. En general, la estructura obtenida se compone de tres macro componentes que se repiten para cada tendencia y un proyecto final para cerrar el curso.

Para cada una de las tendencias se consideraran tres pasos: (i) el estudiante obtendrá los conocimientos teóricos de cada tendencia apoyándose en recursos como el texto guía, vídeos , simuladores y retos para adquirir las bases biológicas e informáticas necesarias; (ii) el estudiante realizará laboratorios para desarrollar las capacidades prácticas necesarias de cada tendencia apoyándose en el *framework* proporcionado; (iii) el estudiante realiza la búsqueda, selección y presentación de artículos de aplicación para entender los usos que se le da a cada algoritmo para solucionar distintos tipos de problema y desarrollar competencias relacionadas a la investigación.

Para el cierre del curso el estudiante debe presentar un proyecto final en el que seleccione un problema y posteriormente diseñe,implemente y presente la solución correspondiente utilizando los conocimientos obtenidos durante el curso. El objetivo de este componente es permitir que el estudiante aplique los conocimientos obtenidos en un problema de su interés además de observar las competencias que cada estudiante obtuvo en el proceso y de esta manera validar si se cumplieron los objetivos del curso.

Para comprobar la efectividad del diseño planteado, se generó una versión piloto de los recursos propuestos y se desplegaron en el ambiente virtual Moodle, esto con el objetivo de permitir que dos estudiantes pudieran probar tanto la primera versión de los recursos, como el diseño general del curso. La retroalimentación obtenida por parte de los estudiantes muestra

la efectividad de la estructura planteada, si bien es necesario iterar sobre algunos recursos para mejorarlos, se observó que la estructura de andamiaje utilizada facilita y motiva el aprendizaje, también se evidenció que el orden de las actividades, junto con los recursos que las acompañan, permiten una experiencia de aprendizaje fluida y agradable para los estudiantes.

En base a los resultados anteriores, consideramos que el diseño generado es efectivo para la enseñanza de computación bioinspirada, y se tendrán en cuenta los comentarios de los estudiantes para mejorar los recursos y analizar puntos de mejora en el diseño.

2.7. Trabajo futuro

A partir de los comentarios de los estudiantes se identificaron los siguientes objetivos para trabajo futuro: (i) analizar la posibilidad de incluir pruebas calificadas automáticamente (pruebas tipo quiz) para que los estudiantes puedan validar el conocimiento teórico obtenido y obtener retroalimentación de manera inmediata; (ii) establecer un esquema de creación de laboratorios para facilitar la inclusión de nuevos laboratorios al curso; y (iii) generar nuevas versiones de los recursos teniendo en cuenta los comentarios recibidos.

Adicionalmente, se plantea la iteración del diseño, haciendo que más estudiantes prueben el curso, para poder tener diversidad de opiniones y más ideas para mejorar.

El propósito es lograr incluir este curso de computación bioinspirada como una asignatura en la Escuela Colombiana de Ingeniería Julio Garavito.

Capítulo 3

Curaduría de recursos audiovisuales

3.1. Introducción

Durante el diseño de un curso de computación bioinspirada se observó que muchos de los estudiantes interesados en esta área no cuentan con los conceptos base de biología necesarios. Partiendo del contexto de un curso introductorio destinado a estudiantes de pregrado, surge la necesidad de hacer que estos conceptos biológicos puedan ser comprendidos con facilidad. Es por ello que se propone el uso de recursos audiovisuales(videos) que muestren y expliquen cada uno de los fenómenos que se quieren enseñar pues se considera que poder visualizar el funcionamiento de estos no solo genera un aprendizaje significativo sino que también puede aumentar el interés de los estudiantes en el área .

Continuando con esta idea y teniendo en cuenta la frase de Donald Knuth, "*An algorithm must be seen to be believed*"[51] también se busca facilitar el aprendizaje a nivel informático, por lo que se propone el uso de simuladores para que los estudiantes puedan experimentar y observar el funcionamiento de los algoritmos antes de implementarlos de modo que al momento de escribir el código ya cuenten con una experiencia previa del funcionamiento del algoritmo y las capacidades de los distintos hiper-parámetros.

A partir de lo anterior,este trabajo busca establecer una metodología para la búsqueda tanto de vídeos como de simuladores, y a partir de dicha metodología realizar la exploración y selección correspondiente con el fin de incluir estos recursos audiovisuales en un curso introductorio de computación bioinspirada.

3.2. Metodología

A continuación se presenta las metodologías que se utilizarán en la curaduría de vídeos y de simuladores para las distintas tendencias de la computación inspirada por la naturaleza

3.2.1. Curaduría de vídeos

Definición de objetivos

El objetivo de la curaduría es seleccionar un vídeo o un conjunto de vídeos que sumen una duración total aproximada de 5 a 10 minutos que permita al estudiante visualizar y comprender, a nivel introductorio, el proceso biológico que inspira a cada una de las tendencias.

En caso de no encontrar vídeos con imágenes reales, se buscará que la visualización sea tan fiel como sea posible a lo biológico.

Elección de lugar y cadenas de búsqueda

La búsqueda se realizará sobre la plataforma de Youtube; por ser una de las plataformas de vídeo con más contenido en la actualidad, se espera encontrar diversos tipos de contenido para suplir la necesidad de un visualizador del proceso biológico de cada una de las tendencias.

Las cadenas de búsqueda se especificarán para cada uno de las tendencias. Adicional a las cadenas definidas también se utilizarán los equivalentes en inglés de estas cadenas de búsqueda, dando prioridad a la búsqueda de contenido en español, esto dado que nuestro objetivo es poder presentar recursos en español a los estudiantes.

También se tendrán en cuenta los vídeos presentados como sugerencias de la plataforma que estén relacionados a los vídeos seleccionados durante la búsqueda

Preselección

La preselección se realizará considerando un conjunto de criterios generales para todas las tendencias y adicionalmente, si es necesario, en cada tendencia se adicionarán algunos criterios específicos. A continuación, se mencionan los criterios generales:

Criterios de inclusión

- Vídeos que muestren y expliquen la estructura y comportamiento biológico de cada enfoque
- Vídeos que permitan visualizar el fundamento biológico de cada enfoque por medio de datos biológicos reales o por medio de simulaciones por computador.

Criterios de exclusión

- Vídeos con duración mayor a 15 minutos
- Vídeos que utilicen conceptos avanzados o demasiado específicos en biología o química para explicar los procesos

Para la preselección se visualizarán los vídeos por completo y se incluirán los que cumplan con todos los criterios de inclusión y que no cumplan con ninguno de los criterios de exclusión.

Selección

Los vídeos seleccionados serán evaluados utilizando los siguientes criterios:

- Interés del contenido (1-5)
- Completitud:(1-5)
- Corrección (1-5)
- Claridad (1-5)
- Método de visualización (1-5)
- Contenido en español (0-1)
- Duración
- Año

La selección del vídeo o vídeos se realizará considerando los resultados de evaluación de los vídeos preseleccionados

Caracterización vídeo deseable

Teniendo en cuenta las métricas de evaluación presentadas anteriormente, a continuación se explicará que se desea obtener en los vídeos seleccionados.

- Interés del contenido: Se busca que los vídeos seleccionados presenten los temas de forma tal que se genere interés en el estudiante y mantenga su atención en el contenido del vídeo.
- Completitud: Se definirán los criterios de completitud para cada tendencia teniendo en cuenta lo que se busca mostrar para cada tema
- Corrección : El contenido de los vídeos debe ser correcto y preciso, la idea de esto es evitar confundir al estudiante con conceptos erróneos.

- Claridad : Los vídeos deben presentar los conceptos de forma clara, se busca lograr que incluso personas sin un amplio conocimiento previo en el área logren entender los procesos biológicos que inspiran cada algoritmo.
- Método de visualización : Se busca que el método de visualización sea lo más fiel a las representaciones reales de los fenómenos, en caso de no ser posible se prefieren los vídeos que presenten simulaciones de los procesos.
- Contenido en español: Los vídeos preferiblemente deben presentar la información en español
- Duración: Los vídeos escogidos no pueden tener una duración mayor a 10 minutos, se espera que la visualización de los vídeos no sea una tarea tediosa para los estudiantes y que puedan obtener mucho conocimiento en poco tiempo a partir de la riqueza de las representaciones gráficas.

3.2.2. Curaduría de simuladores

Definición de objetivos

El objetivo de esta curaduría es seleccionar un simulador que permita al estudiante explorar el funcionamiento de los algoritmos y experimentar con los distintos parámetros e hiper-parámetros de cada una de las tendencias.

Es importante que el simulador permita la solución de varios problemas para evidenciar la necesidad de sintonización de hiper-parámetros adecuados para cada problema.

Adicionalmente se busca que el simulador sea intuitivo y fácil de usar para los estudiantes.

Elección de lugar y cadenas de búsqueda

La búsqueda se realizará directamente sobre Google, las cadenas de búsqueda se especificarán para cada tendencia. Es importante resaltar que la búsqueda se enfoca en cadenas en inglés, aunque también se utilizarán los equivalentes en español.

Puesto que por revisiones anteriores sabemos de la escasez de este tipo de herramientas en español teniendo en cuenta que lo más probable es que se obtenga un simulador en inglés, el aspecto de la facilidad de uso y la experiencia de usuario toma un punto importante en los criterios de selección.

Preselección

Se tienen un conjunto de criterios generales para todos las tendencias y adicionalmente, si es necesario, en cada una se detallarán algunos criterios específicos.

A continuación, se mencionan los criterios generales

Criterios de inclusión

- Simuladores disponibles de forma gratuita
- Simuladores que permitan visualizar el proceso del algoritmo, los parámetros y el efecto de los hiper-parámetros.
- Simuladores que permitan la experimentación con hiper-parámetros.
- Simuladores que permitan solucionar varios problemas.

Criterios de exclusión

- Simuladores que requieran de la modificación de código para realizar experimentos

Para la preselección se utilizarán los simuladores encontrados y se incluirán los que cumplan con todos los criterios de inclusión y con ninguno de los criterios de exclusión

Selección

La selección de los simuladores se realizará con base en los siguientes criterios.

- Sintonización de hiper-parámetros (1-5)
- Visualización de Parámetros (1-5)
- Método de visualización (1-5)
- Usabilidad (1-5)
- Cantidad de problemas que permite solucionar (1-5)

Para la evaluación se realizará una exploración detallada de las funcionalidades que cada uno de los simuladores preseleccionados.

Caracterización de simulador deseable

Teniendo en cuenta las métricas de evaluación presentadas anteriormente, a continuación se explicará por cada métrica que se desea obtener en los simuladores

- **Parámetros**: Se definirá para cada tendencia
- **Hiper-Parámetros**: Se definirá para cada tendencia
- **Visualización**: El simulador debe presentar toda la información del proceso de forma clara y ordenada, dentro de esta información se incluye de manera general:
  - Hiper-parámetros disponibles para experimentar
  - Problemas disponibles para solucionar
  - Gráficas que muestren la evolución de la solución

  Adicionalmente, para cada tendencia se especificará información relevante para cada algoritmo.
- **Problemas que permite solucionar**: Es importante que el simulador cuente con varios problemas para solucionar, puesto que esto lleva a que los estudiantes entiendan la importancia de las representaciones y de la sintonización de hiper-parámetros.
- **Usabilidad**: El simulador debe ser fácil e intuitivo de usar, adicionalmente debe poderse ejecutar correctamente en la mayoría de equipos y debe poder ejecutarse sin conocimientos de algún lenguaje en especifico (debe ser un cliente web o contar con un instalador)

3.3. Curaduría

3.3.1. Curaduría de vídeos

Teniendo en cuenta la metodología general curaduría de vídeos, a continuación, se presenta el proceso realizado para cada una de las tendencias

Computación neuronal: Redes neuronales

Elección de cadenas de búsqueda Las cadenas de búsqueda utilizadas para encontrar vídeos de redes neuronales son las siguientes:

- Redes neuronales humanas
- Funcionamiento de la neurona
- Conexiones neuronales
- Visualización funcionamiento de la neurona
- Funcionamiento de la neurona
- Simulación sinapsis
- Sinapsis y aprendizaje

Preselección Adicional a los criterios generales se incluye el siguiente criterio de inclusión:

Criterios de inclusión específicos

- Vídeos que permitan entender la relación entre la sinapsis y el aprendizaje/ memoria

Se encontraron una gran cantidad de vídeos relacionados a los temas de búsqueda, se visualizaron 25 vídeos con los que se realizó la preselección de la que finalmente se escogieron 6 en la preselección.

Selección Para la evaluación se completitud se utilizarán los siguientes criterios:

- Detalles del proceso biológico (1-5): el vídeo debe detallar la estructura y funcionamiento de la neurona y las redes neuronales así como del proceso biológico detrás de la sinapsis
- Detalles sobre aprendizaje (1-5): es necesario detallar el papel de las neuronas en el aprendizaje y la memoria del ser humano y la relación entre las conexiones neuronales(sinapsis) y la obtención de nuevos conocimientos

Los resultados de la evaluación realizada se presentan en la tabla 3.1

Titulo	Claridad	Veracidad	Interés	Detalles sobre aprendizaje	Método de visualización	Español?	Detalles biológicos	Duración	Año
Las neuronas [67]	4.5	5	3	2	3	1	2	02:00	2013
Aprendizaje, Memoria y Cerebro [66]	4.5	4.5	3.5	3.5	2	1	3	03:39	2012
¿Cómo aprendemos? Aprendizaje y conexiones neuronales.[85]	5	5	4	4	4	1	3	10:47 (Fragmento util 0:00 - 3:55)	2019
Neurona y Aprendizaje [8]	5	4.5	4	4.5	4	1	3.5	02:51	2019
Experiencia 360° | Sinapsis, el cerebro por dentro [70]	4.5	5	4	2	4.5	1	4	03:43	2019
How a synapse works [45]	5	5	4	1	3.5	0	4.5	05:01	2017

Cuadro 3.1: Evaluación de vídeos de red neuronal

En general se observó que los vídeos suelen tener o un enfoque hacia los detalles biológicos o un enfoque hacia los detalles de cómo funcionan las redes neuronales y el aprendizaje, pero rara vez contenían ambos componentes.

Teniendo en cuenta la evaluación y la observación presentada anteriormente, el vídeo más equilibrado es "Neurona y Aprendizaje" [8]. Sin embargo, consideramos que para dar un conocimiento completo al estudiante lo mejor es optar por seleccionar dos vídeos: el primero es el

vídeo “Neurona y Aprendizaje”, en el que se explica lo relacionado a la creación de conexiones y el aprendizaje, dando algunos detalles de los procesos biológicos involucrados; el segundo es el vídeo “ Experiencia 360° |Sinapsis, el cerebro por dentro”[70] donde se muestran y explican los detalles biológicos del proceso de una forma clara y con un método de visualización que incluso permite cierto nivel de interacción. Los dos vídeos suman 6 minutos 34 segundos, lo que cumple con el tiempo propuesto en los objetivos.

Computación evolutiva: Algoritmos genéticos

Elección de cadenas de Búsqueda Las cadenas de búsqueda utilizadas para encontrar vídeos de la teoría biológica relacionada a los algoritmos genéticos son las siguientes:

- Selección natural
- Genética y evolución
- Simulación de genética mendeliana
- Simulación de selección natural

Preselección Adicional a los criterios generales se incluyen los siguientes criterios de inclusión:

- Vídeos que expliquen los conceptos de genética, evolución y selección natural

Se encontraron y visualizaron 27 vídeos relacionados a las cadenas de búsqueda seleccionadas, la mayoría de estos correspondientes al tema de la selección natural. De estos vídeos se preseleccionaron 10 al aplicar los criterios de selección definidos. En general se encontró material diverso y de muy buena calidad. Un elemento común en la mayoría de vídeos encontrados es el uso de animaciones, creemos que esto se debe a que este material es útil para estudiantes de colegio y que esta forma de visualización puede ayudar a captar su atención y facilitar la explicación.

Selección Para la evaluación de completitud se utilizarán las siguiente métricas:

- Detalles sobre genética (1-5) : el vídeo debe explicar las bases genéticas que sustentan el algoritmo, entre esto se debe incluir la explicación del cruce genético realizado en la reproducción,la aparición de rasgos hereditarios y las mutaciones.
- Detalles sobre selección natural y evolución (1-5): el vídeo debe explicar el proceso de selección natural,mostrando como a partir de las condiciones del ambiente se guía la evolución de una especie al incentivar el predominio de individuos con los rasgos que permitan mayor adaptación a las condiciones y por lo tanto mayor supervivencia.

Los resultados de la evaluación realizada se presentan en la tabla 3.2

Titulo	Claridad	Veracidad	Interés	Detalles sobre genética	Método de visualización	Español?	Detalles sobre selección natural y evolución	Duración	Año
Explicación sobre el simulador de selección natural[26]	3.5	4	3.5	3	4.5	1	3.5	05:50	2020
Selección natural y adaptación | HHMI BioInteractive Video [14]	5	5	5	3	5	1	5	10:29	2017
Las teorías de Darwin[44]	4.5	5	4	3	3.5	1	4	03:20	2016
Ciencia express: selección natural[29]	4.5	5	4.5	3	4	1	4	03:30	2014
¿Qué es la Selección Natural? - What is Natural Selection?[38]	4	5	4.5	3	4	1	3.5	02:00	2020
Explicación biológica genética de la evolución de las especies[53]	5	5	4.5	4.5	4.5	1	3	01:56	2015
¿Por qué nos parecemos a nuestros papás? La Genética - CuriosaMente 161 [24]	4.5	5	4	5	4	1	1	07:47	2019
How Mendel´s pea plants helped us understand genetics[36]	4.5	5	4	5	4	0	1	02:55	2013
Simulando la selección natural[71]	3.5	4.5	4.5	3	4.5	0	3	10:00	2018
simulacion seleccion natural[11]	4	4.5	3.5	3.5	4.5	1	4	(2:03-9:27)	2019

Cuadro 3.2: Evaluación de vídeos de algoritmos genéticos

Para abarcar todos los conceptos teóricos que se desean trasmitir se decidió tomar dos vídeos que se complementen, los vídeos seleccionados son : (1) "Explicación biológica genética de la evolución de las especies" [53] que abarca la explicación a nivel genético que da paso a la evolución utilizando un método de simulación para mostrar y explicar el proceso genético de forma clara, adicionalmente introduce conceptos de evolución y selección natural.(2) "Ciencia exprés: selección natural"[29] que detalla el proceso de selección natural por medio de un ejemplo usando animaciones que representan la evolución de una raza de conejos bajo ciertas condiciones de ambiente, menciona algunos de los componentes de genética pero no entra al detalle como en el primer vídeo, si bien su forma de visualización no es tan fiel a la realidad presenta el tema de una manera clara y didáctica que facilita el entendimiento del fenómeno. Con estos inspira a los algoritmos genéticos. Los dos vídeos suman 5 minutos 26 segundos, lo que cumple con el tiempo propuesto en los objetivos.

Computación de enjambre: Algoritmo de optimización de colonia de hormigas

Elección de cadenas de búsqueda Las cadenas de búsqueda utilizadas para encontrar vídeos del comportamiento de búsqueda de comida de las hormigas son las siguientes:

- Hormigas encontrando caminos óptimos
- Como se comunican las hormigas?
- Como funciona la colonia de hormigas?
- Charlas TED: colonia de hormigas
- Hormigas búsqueda de comida
- Experimento del puente binario con hormigas

Preselección Adicional a los criterios generales se incluye el siguiente criterio de inclusión:

Criterios de inclusión específicos

- Vídeos que permitan entender el uso que le dan las hormigas a las feromonas para comunicar los mejores caminos hacia la comida

Se encontraron y visualizaron 15 vídeos relacionados a las cadenas de búsqueda seleccionadas,se encontraron muchos vídeos relacionados a otros comportamientos de las hormigas, pero relativamente pocos al comportamiento deseado. De estos vídeos se preseleccionaron 7 al aplicar los criterios de selección definidos.

Selección Para la evaluación se completitud se utilizarán los siguientes criterios:

- Detalles del uso de feromonas (1-5) : el vídeo debe mostrar como las hormigas hacen uso de las feromonas para comunicarse y tomar decisiones.
- Detalles sobre la construcción de caminos óptimos (1-5) : se desea que el video muestre claramente la capacidad que tienen las colonias de hormigas de construir caminos óptimos desde la colonia hacia las fuentes de comida y la relación que tiene la elección de este camino con la deposición de feromonas.

Los resultados de la evaluación realizada se presentan en la tabla 3.3

Titulo	Claridad	Veracidad	Interés	Detalles del uso de feromonas	Método de visualización	Español?	Detalles sobre la construcción de caminos óptimos	Duración	Año
¿Como Encuentran tu Comida las Hormigas?| CURIOSIDADES[37]	3	5	2.5	4	2.5	1	1	fragmento: 0:00-3:07	2019
TED: Dentro de la colonia de hormigas [56]	4.5	5	3.5	4.5	3.0	1	4	04:49	2016
Lo que verías si pudieras entrar a una colonia de hormigas[42]	4.5	4	4	4	3.0	1	1	fragmento: 0:00-2:00	2021
COMO HABLAN LAS HORMIGAS? Reclutan para Cazar - Comunicación y Lenguaje Feromonas Pheidole Pallidula[47]	4.5	5	5	5	5	1	1	fragmento: 9:55-15:20	2020
Algoritmo de hormigas para encontrar el camino óptimo. 1/2[75]	3.5	4	3.5	2.5	3.5	1	3.5	05:56	2020
Algoritmo de hormigas para encontrar el camino óptimo. 2/2[76]	3.5	4	3.5	2.5	3.5	1	3.5	02:40	2020
Can ants choose the shortest path to a food source?[12]	4.5	5	5	4	5	0	5	fragmento: 1:28-5:18	2020

Cuadro 3.3: Evaluación de vídeos de algoritmo de optimización de colonia de hormigas

Se seleccionó el vídeo "*Can ants choose the shortest path to a food source?*"[12] puesto que es uno de los vídeos con mejores valores en las métricas seleccionadas, y de los pocos que menciona las feromonas y la capacidad de las hormigas para construir caminos óptimos. Este vídeo es muy interesante porque permite visualizar como las hormigas son capaces de crear caminos óptimos a partir de un experimento con hormigas reales; sin embargo, el vídeo está en inglés y por esto no cumple con una de las características deseables planteadas, para solucionar esto se realizó un trabajo adicional en el que se colocaron subtítulos al vídeo dado que los subtítulos generados automáticamente no eran claros en algunas secciones del vídeo. Adicionalmente, se decidió recortar una parte del vídeo puesto que correspondía a diapositivas introduciendo el tema usando únicamente texto. Con esto se obtiene un vídeo de 4 minutos por lo que se cumple con el objetivo de tiempo planteado.

3.3.2. Curaduría de simuladores

Teniendo en cuenta la metodología general de búsqueda de simuladores, a continuación, se presentan las búsquedas relacionadas a cada uno de las tendencias

Computación neuronal: Redes neuronales

Elección de cadenas de búsqueda Las cadenas de búsqueda utilizadas para encontrar simuladores de redes neuronales son las siguientes:

- *Neural network online simulator*
- *Interactive visualization neural network*
- *Fully connected neural network visualization*

Adicionalmente se revisó un simulador que se encontró durante una revisión previa en un curso ofrecido en una página web.

Preselección Adicional a los criterios generales se incluye el siguiente criterio de inclusión:

Criterios de inclusión específicos

- Simuladores que muestren el funcionamiento de una red neuronal completamente conectada (*Fully-Connected*)

En la búsqueda se evidenció la ausencia de material en español, aún en inglés se encontraron sólo 7 simuladores, algunos descartados por estar relacionados a redes convolucionales y otros por no permitir la experimentación con hiper-parámetros. Finalmente se seleccionaron 4 simuladores.

Selección Las características específicas deseadas, que se unen a las definidas en la metodología general, se mencionan a continuación:

- **Parámetros**: El simulador debe permitir la visualización clara de los parámetros de peso y sesgo de la red.
- **Hiper-Parámetros**: El simulador debe permitir la experimentación con distintas arquitecturas de red, con el ritmo de aprendizaje,las funciones de activación , la cantidad de iteraciones y la proporción entre datos de prueba y validación.
- **Visualización**: La información específica que se debe mostrar se detalla a continuación
  - Parámetros de peso y sesgo
  - Arquitectura de la red
  - Datos de prueba y de validación
  - Gráfica del dataset usado (Para casos 2D)
  - Estado de la solución (Que tan bien se ajusta el modelo a los datos)

Los resultados de la evaluación realizada se presentan en la tabla 3.4

Nombre	Hiperparámetros	Parámetros	Visualización	Usabilidad	Problemas que permite solucionar
Tensorflow Playground[77]	4.5	4	5	5	3.5
Step-by-step Artificial Neural Network visualizer[6]	3.5	3.5	3.5	3.5	4.5
Neural Network demo[72]	4	3.5	4	4	4.5
MultiLayer Perceptron[20]	3	1	3	3	2

Cuadro 3.4: Evaluación de simuladores de red neuronal

De esta evaluación resultó seleccionado el simulador "*Tensorflow playground*"[77]. En la evaluación vemos que es el que tiene mejor valoración a nivel general: presenta toda la información relevante del algoritmo de forma clara y ordenada, es fácil e intuitivo de usar, está disponible para usar desde el navegador, es posible visualizar e incluso cambiar manualmente los valores de los parámetros y permite experimentar con todos los hiper-parámetros deseados para esta tendencia. Su único punto débil es que solo permite experimentar con un número limitado de problemas; sin embargo, los *datasets* que presenta permiten tener un buen entendimiento del funcionamiento de las redes neuronales completamente conectadas y sus hiper-parámetros más importantes.

Computación evolutiva: Algoritmos Genéticos

Elección de cadenas de búsqueda Las cadenas de búsqueda utilizadas para encontrar simuladores de algoritmos genéticos son las siguientes:

- *genetic algorithms [simulator |demo |sandbox]*
- *[learning |teaching] tool for genetic algorithm*
- *[netlogo |javascript] genetic algorithms*

Preselección Adicional a los criterios generales se incluyen los siguientes criterios de inclusión:

Criterios de inclusión específicos

- Simuladores que muestren el funcionamiento de algoritmos genéticos
- Simuladores que hagan explícita la representación utilizada para solucionar el problema
- Simuladores que usen representaciones simples como listas de binarios, de enteros o de caracteres.

El material encontrado para este enfoque es muy escaso, si bien se encontraron varios artículos que muestran simuladores relacionados, estos ya no se encuentran disponibles para ser usados. Se encontraron 8 simuladores disponibles para experimentar, de estos se seleccionaron 5 para la evaluación.

Selección Las características específicas deseadas, que se unen a las definidas en la metodología general, se mencionan a continuación:

- **Parámetros**: El simulador debe permitir la visualización clara de las representaciones utilizadas en cada problema, además del estado de los individuos en cada iteración.

- **Hiper-Parámetros**: El simulador debe permitir la experimentación con distintos operadores de selección, cruce y mutación además de los hiper-parámetros requeridos por cada operador como por ejemplo la taza de reproducción, la taza de mutación, tamaño de la población, entre otros.
- **Visualización**:La información específica que se debe mostrar se detalla a continuación:
  - Estado actual de la población (o al menos el estado de los individuos seleccionados en cada iteración)
  - Proceso de cruce y mutación
  - Mejor, peor y promedio de la población con respecto a la función de evaluación

Los resultados de la evaluación realizada se presentan en la tabla 3.5

Titulo	Hiper-parámetros	Visualización	Parámetros	Usabilidad	Problemas que permite solucionar
2D Genetic Algorithm [17]	4	2.5	2	4	1
Simple Genetic Algorithm [80]	4	3.8	2	4	1
Robby the Robot [62]	3.5	3.5	2	2	1
Minimal Genetic Algorithm[54]	4	3.8	3.5	4	1
Genetic Cars [57]	3.5	3.5	2	4	1

Cuadro 3.5: Evaluación de simuladores de algoritmos genéticos

De esta evaluación resultó seleccionado el simulador "Minimal Genetic Algorithm"[54] por ser el más equilibrado de los 5 con respecto a las métricas seleccionadas, sin embargo no consideramos que este simulador cumpla con las expectativas planteadas,puesto que: no permite experimentar con los operadores genéticos, la forma en la que se muestran los cruces y mutaciones no es clara y solo permite solucionar un problema muy simple. Por esto se considera valioso el desarrollo de un trabajo futuro que construya un simulador que cumpla con las especificaciones deseables.

Caracterización simulador deseable

Computación de enjambres: Algoritmo de optimización de colonia de hormigas

Elección de cadenas de búsqueda Las cadenas de búsqueda utilizadas para encontrar simuladores del algoritmo de optimización de colonia de hormigas son :

- *Ants colony optimization [simulator |demo |sandbox]*
- *[learning |teaching] tool for ants colony optimization*
- *[netlogo |javascript] Ants colony optimization*

Preselección Adicional a los criterios generales se incluyen los siguientes criterios de inclusión y exclusión:

Criterios de inclusión específicos

- Simuladores que muestren los rastros de feromona dejados por las hormigas
- Simuladores que muestren los caminos tomados por las hormigas

Criterios de exclusión específicos

- Simuladores relacionados a mostrar el comportamiento biológico de las hormigas (simulación de la naturaleza)

El material encontrado para este enfoque es muy escaso, si bien se encontraron varios artículos que muestran simuladores relacionados, estos ya no se encuentran disponibles para ser usados, otros simuladores encontrados en la web presentaban problemas al ser ejecutados y otros representaban la simulación del comportamiento biológico de las hormigas y no del algoritmo en sí, se logró experimentar con 7 simuladores, de los cuales se seleccionaron 4 para la evaluación

Selección Las especificaciones deseables se mencionarán a continuación:

- **Parámetros**: El simulador debe permitir la visualización clara de los caminos que toman las hormigas, de los rastros de feromona y la mejor solución de cada iteración
- **Hiper-Parámetros**: El simulador debe permitir la experimentación de los hiper-parámetros de tasa deposición feromonas (α), peso de la heurística de deseabilidad(β), tasa de evaporación (ρ), número de individuos y número de iteraciones.
- **Visualización**: El simulador debe presentar toda la información del proceso de forma clara y ordenada, dentro de esta información se incluye:
  - Estado de las feromonas en cada iteración
  - Caminos seleccionados por las hormigas
  - Nodos del grafo del problema
  - Estado inicial y final deseado (Si aplica al problema)

Los resultados de la evaluación realizada se presentan en la tabla 3.6

Titulo	Hiperparámetros	Visualización	Parámetros	Usabilidad	Problemas que permite solucionar
Visualisation of Ant Colony Optimisation[69]	4.5	4	3.5	4	1
JS ACO[32]	5	3	4	3.5	1
Ant Colony Optimization For Hackers[48]	5	4	4	4	1
ACO Simulator[40]	4.5	4.5	4.5	4.5	1

Cuadro 3.6: Evaluación de simuladores de algoritmos de optimización de colonia de hormigas

De esta evaluación resultó seleccionado el simulador "*ACO Simulator*"[40], es de los simuladores más equilibrados, es fácil de usar y muestra claramente los rastros de feromona además de la información de como ha evolucionado el problema a lo largo de las generaciones; en cuanto a hiper-parámetros, permite sintonizar α ,β , la evaporación y permite controlar las iteraciones. Como puntos débiles encontramos que no permite la sintonización del número de individuos y al igual que todos los otros simuladores encontrados, únicamente permite solucionar el problema de TSP [1].

[1]Travelling Salesman Problem

Si bien el simulador permite entender el funcionamiento del algoritmo, sería interesante el desarrollo de un simulador que permita solucionar otros problemas a partir de configuraciones de grafos dados, para que los estudiantes puedan experimentar con el algoritmo sobre variedad de escenarios.

3.4. Conclusiones

En este trabajo se logró la definición de metodologías generales de curaduría tanto para vídeos como para simuladores, teniendo en cuenta la necesidad de estos dos recursos de aprendizaje para la enseñanza de las distintas tendencias de la computación inspirada por la naturaleza. Utilizando estas metodologías se seleccionaron 5 vídeos y 3 simuladores.

Los vídeos encontrados cumplieron con las expectativas planteadas, se considera que los conjuntos de vídeos seleccionados facilitan el entendimiento de los conceptos biológicos al mismo tiempo que generan mucho interés en los estudiantes hacia el área.

En cuanto a los simuladores seleccionados, se considera que los simuladores de computación neuronal e inteligencia de enjambres cumplen con las expectativas planteadas, al permitir la experimentación de los distintos componentes de cada algoritmo de una forma simple e intuitiva; por el contrario, no fue posible encontrar un simulador de algoritmos genéticos con esas características, el simulador seleccionado permite la experimentación parcial de los componentes del algoritmo y en general es más difícil de usar que los otros simuladores seleccionados.

3.5. Trabajo Futuro

Teniendo en cuenta que no fue posible encontrar un simulador de algoritmos genéticos que cumpliera con los requisitos planteados, se propone como trabajo futuro un desarrollo propio de de este simulador que cumpla con todos los requisitos planteados en la metodología, y de esta forma poder presentar recursos de excelente calidad al estudiante para todas las tendencias planteadas.

Capítulo 4

Framework

4.1. Introducción

En el desarrollo de este proyecto se realizó el diseño e implementación de una serie de *frameworks* con el objetivo de apoyar la implementación de modelos de computación natural con los cuales solucionar problemas de forma practica e intuitiva, esto mediante laboratorios que el estudiante realizará.

Por cada tendencia que se abarca en el curso se desarrolló un *framework* el cual ofrece la implementación del algoritmo más representativo de la tendencia correspondiente. El desarrolló de estos se realizó de forma iterativa donde en cada iteración se realizaban mejoras a diseño según fuera el caso.

Un *framework* es un producto de software que ofrece una serie de componentes o módulos de software los cuales definen una estructura o arquitectura especifica, dicha estructura define la forma básica o genérica en la que este programa debe funcionar por lo que en la mayoría de casos es necesario que el programador que utilice el *framework* tenga que extenderlo usando los componentes básicos. De esta manera se implementan los componentes concretos del *framework* según la problemática que se quiera resolver. En resumen un *framework* mediante su arquitectura y diseño expone una metodología de trabajo, esta metodología está dada por la posibilidad de extender o utilizar los distintos componentes que conforman la arquitectura expuesta.[73]

4.2. Metodología

Como se mencionó se desarrolló un *framework* por cada tendencia el cual ofrece la implementación de un modelo correspondiente al algoritmo que se seleccionó para cada una de ellas.

Los *frameworks* correspondientes a cada sección o tendencia presente en el curso son los siguientes:

- Computación neuronal: *framework* redes neuronales
- Computación evolutiva: *framework* algoritmo genético
- Inteligencia de enjambre: *framework* algoritmo de optimización por colonia de hormigas

Estos *frameworks* ofrecen los componentes necesarios para implementar de forma sencilla el algoritmo correspondiente, permitiendo la posibilidad de usar directamente o extender cada componente. De esta manera, mediante el uso del *framework* se construye un modelo capaz de solucionar el problema de interés, dicho modelo puede ser sintonizado con el objetivo de encontrar aquellos hiper-parámetros que representan el mejor desempeño.

El proceso de desarrollo de los *frameworks* se realizó de forma iterativa siguiendo el proceso clásico dado por el ciclo de vida de software, dicho proceso se realizó por cada uno de los *frameworks*.

Primero se especificó el objetivo de este, la estructura y los servicios que debía ofrecer, luego se realizó el diseño y la codificación, seguido por las pruebas. Una vez terminado este ciclo inicial puede volver a iniciar según las revisiones correspondientes, de esta manera al final se logra de forma ágil un producto de software que es sólido y completo. En la figura 4.1 se observa el proceso de desarrollo seguido.

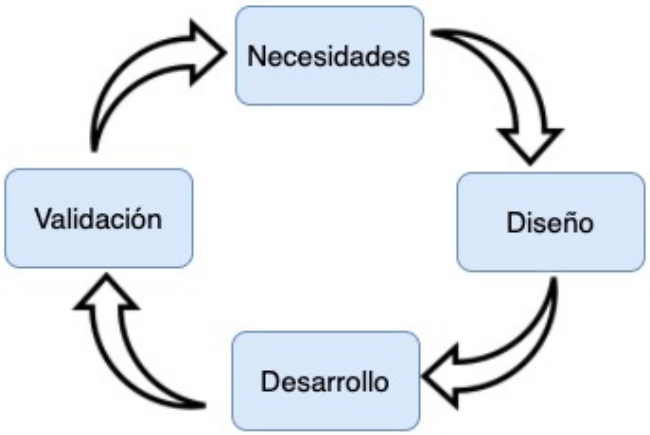


Figura 4.1: Ciclo desarrollo de software

4.3. Resultados

Se construyeron tres *frameworks* que implementan la meta-heurística del algoritmo más representativo por cada tendencia. En las siguientes secciones se detallará en profundidad en cada uno de estos.

Computación evolutiva: *framework* algoritmo genético Inteligencia de enjambre: *framework* algoritmo de optimización por colonia de hormigas

4.3.1. Computación neuronal: *framework* redes neuronales

Este *framework* facilita el diseño y construcción de un modelo de red neuronal multicapa con propagación hacia atrás. Estos modelos de redes neuronales pueden usarse para resolver problemas de clasificación y regresión por lo que son ampliamente estudiados y usados para resolver una gran variedad de problemas.

Este *framework* permite la construcción de una red neuronal de forma sencilla. De esta manera el estudiante puede solucionar problemas de forma sencilla, práctica y rápida obteniendo buenos resultados.

Este *framework* presenta una serie de componentes lo cuales ofrecen distintas implementaciones, permitiendo de esta manera el uso de implementaciones distintas, construyendo modelos con varias arquitecturas y experimentar sobre estas.

Diseño En la figura 4.2 se observa el diseño correspondiente al *framework*, en este se observan los distintos componentes y como estos se relacionan formando así una estructura desacoplada y limpia, dicha arquitectura permite tener un manejo flexible de los distintos componentes los cuales pueden ser reemplazados o extendidos según la necesidad.

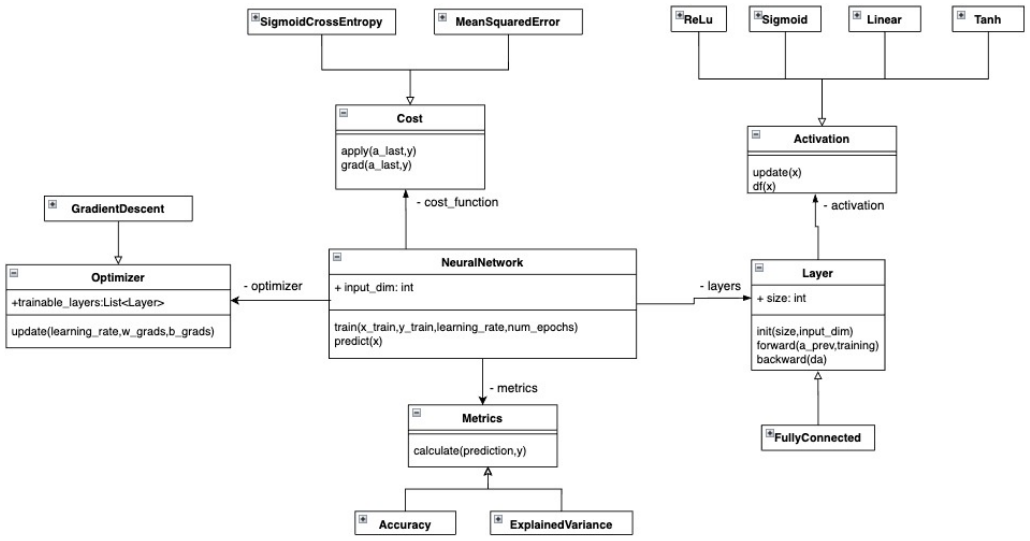


Figura 4.2: Diseño *framework* redes neuronales

activation en Layer debe quedar como rol. Olvidaste quitarla en la clase
A continuación se explican los componentes correspondientes:

- Componente *Layer*: Este componente define una capa de la red neuronal donde sus atributos son la cantidad de neuronas y la función de activación correspondiente a las neuronas en dicha capa. Este componente además define la forma en la que se realiza tanto la propagación adelante como la propagación hacia atrás. La implementación ofrecida corresponde al tipo de capa totalmente conectada.
- Componente *Activation*: Este componente define la función de activación que va a tener una capa, aquí se define tanto la función como su derivada, de esta manera se obtendría los valores necesarios para la propagación adelante y hacia atrás. Las implementación ofrecidas corresponden a las funciones de activación *Lineal*, *ReLu*, *Sigmoide* y *Tanh*.
- Componente *Cost*: Este componente define la función de costo la cual evaluará la salida de la red neuronal, aquí se define tanto la función de costo como su derivada, esta ultima se usaría para realizar la propagación hacia atrás de los errores y actualizar los pesos de acuerdo a la estrategia de optimización que se use. Las implementaciones ofrecidas corresponden a los costos de entropía cruzada (*SigmoidCrossEntropy*) y error cuadrático medio (*MeanSquareError*).
- Componente *Optimizer*: Este componente define el optimizador de la red, siendo este el responsable de actualizar los pesos y de esta forma ir disminuyendo el error obtenido por la red. La implementación ofrecida es el optimizador de descenso del gradiente ((*GradientDescent*).
- Componente *Metrics*: Este componente ofrece la forma de evaluar el desempeño del modelo comparando los datos de salida predichos con los reales, aqui se ofrecen dos implementaciones, *ExplainedVariance* y *Accuracy*.
- Componente *NeuralNetwork*: Este componente define el modelo de red neuronal donde se especifican la dimensión de los datos de entrada y junto con los componentes anteriores

la red neuronal es entrenada, además el modelo una vez entrenado puede predecir valores de salida dado unos valores de entrada.

Validación Una vez diseñado y construido el *framework* este se utilizó para resolver un problema de predicción numérica conocido como *Boston Houses*. Este problema consiste en predecir el precio de un inmueble dado una serie de características y propiedades de este mismo.

El modelo se construyó sin la necesidad de extender el *framework*. La red construida obtuvo buenos resultados donde la varianza explicada por el modelo respecto a los daros de prueba fue de 0.87 lo cual representa que el modelo está en la capacidad de realizar predicciones con un error bajo comparándolo con los datos reales.

4.3.2. Computación evolutiva: *framework* algoritmo genético

Este *framework* facilita el diseño y construcción de un modelo de algoritmo genético, este tipo de algoritmos o modelos están incluidos dentro de los algoritmos evolutivos: los cuales inspirándose en el comportamiento presente en la teoría evolutiva buscan resolver problemas de optimización y combinatoria generalmente. Este *framework* facilita la implementación y construcción de un algoritmo genético donde gracias a una serie de componentes ofrecidos el diseño y construcción de un modelo se haría de forma sencilla permitiendo al estudiante implementar y usar el modelo de forma rápida. De esta manera el estudiante puede solucionar problemas de una forma practica obteniendo buenos resultados.

Diseño Este *framework* presenta una serie de componentes lo cuales ofrecen distintas implementaciones, permitiendo de esta manera la selección de componentes específicos según el problema a resolver. En la figura 4.3 se observa el diseño correspondiente al *framework* donde se observan sus distintos componentes y como estos se relacionan, en dicho diseño se observa como los componentes están desacoplados lo que permite su extensión y flexibilidad al usarlo.

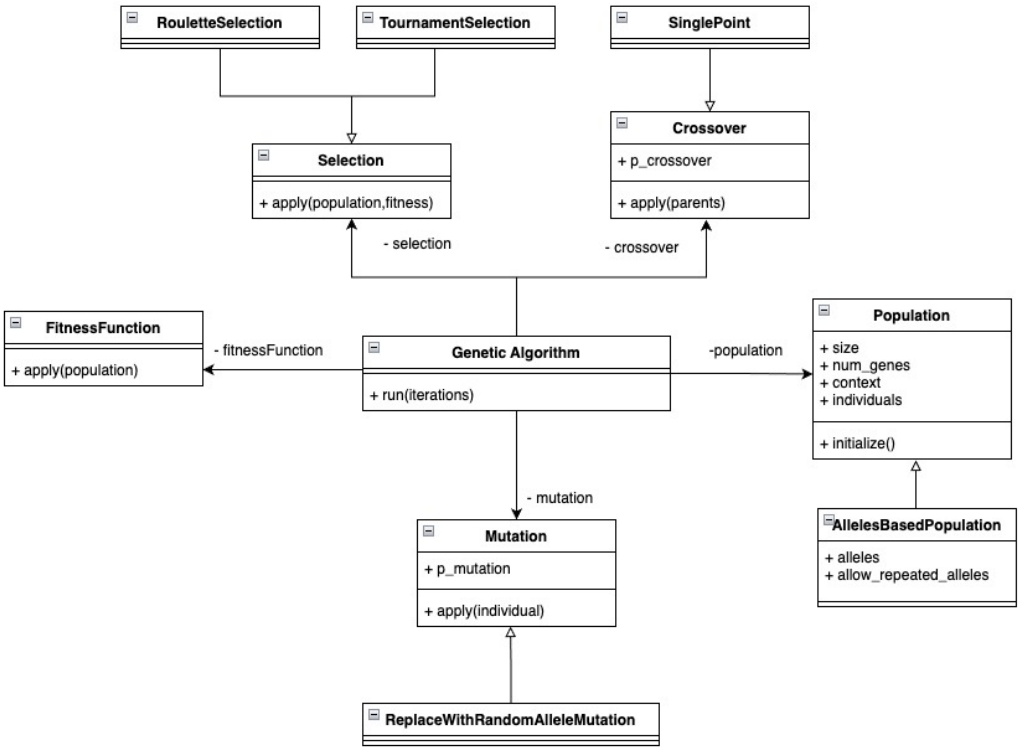


Figura 4.3: Diseño *framework* algoritmo genético

A continuación se presentan los componentes correspondientes.

- Componente *Population*: Este componente representa la población de un modelo de algoritmo genético, aquí un la población tiene un numero de individuos los cuales representan una solución al problema. La implementación ofrecida corresponde a una población donde los individuos son generados según alelos que definen los posibles valores que un individuo puede tener (*AllelesBasedPopulation*).
- Componente *FitnessFunction*: Este componente define la función de aptitud, dicha función es la encargada de evaluar los individuos de la población indicando que tan buenas soluciones estos representan.
- Componente *Selection*: Este componente representa el operador de selección, dicho operador indica la forma en la que se realiza la selección de los individuos que posteriormente se cruzarán. Las implementaciones ofrecidas corresponde a las estrategias de selección por ruleta (*RouletteSelection*) y torneo (*TournamentSelection*).
- Componente *Crossover*: Este componente representa el operador de cruce, este operador indica la manera en la que los individuos se cruzarán, generando de esta manera la siguiente generación de individuos, dicho cruce se realiza según una probabilidad. La implementación ofrecida corresponde a la estrategia de cruce de un punto (*SinglePoint*).
- Componente *Mutation*: Este componente representa el operador de mutación, aquí se realiza la mutación sobre los individuos de la población, esto según una probabilidad. Dicha mutación involucra el cambio sobre los valores de un individuo. La implementación ofrecida corresponde a una estrategia donde se realiza el cambio de forma aleatoria según

los posibles valores que puede tomar un alelo (*ReplaceWithRandomAllleMutation*).
- Componente *GeneticAlgorithm*: Este componente, junto con los demás componentes, define el modelo del algoritmo genético. En este componente se especifican el numero de iteraciones y ofrece un método que se ejecuta proporcionando al final la mejor solución hallada hasta el momento para el problema.

Validación Una vez diseñado y construido el *framework* este se utilizó para resolver un problema conocido como el problema del agente viajero. El objetivo de este problema es encontrar la ruta más corta posible que conecta una conjunto de ciudades, esto dado las posiciones de cada ciudad. Este problema se considera como un problema NP-Hard, por lo que es muy importante en ciencias de la computación ya que resolverlo mediante algoritmos tradicionales es inviable al tener una complejidad alta.

En la construcción de este modelo fue necesario extender la función de aptitud y de esta manera diseñar una función que se adecuara al problema (*TSPFitnessFunction*). Una vez construido el modelo de algoritmo genético correspondiente este alcanzo buenos resultados resolviendo casos complejos del problema del agente viajero.

4.3.3. Computación de enjambre: *framework* algoritmo de optimización de colonia de hormigas

Esta sección corresponde al *framework* del algoritmo de optimización por colonia de hormigas. Este tipo de algoritmos o modelos buscan resolver problemas de búsqueda o optimización inspirándose en el comportamiento de las colonias de hormigas al momento de buscar su comida. Este *framework* facilita la implementación y construcción de este tipo de algoritmos gracias a una serie de componentes ofrecidos las cuales pueden ser extendidos si se requiere. De esta manera el estudiante puede solucionar problemas de una forma practica obteniendo buenos resultados.

Diseño En la figura 4.4 se observa el diseño correspondiente al *framework* donde se observan sus distintos componentes y sus relaciones. La implementación de un modelo de algoritmo de optimización de colonia de hormigas se facilita al usar los componentes dados y extender solamente los componentes necesarios.

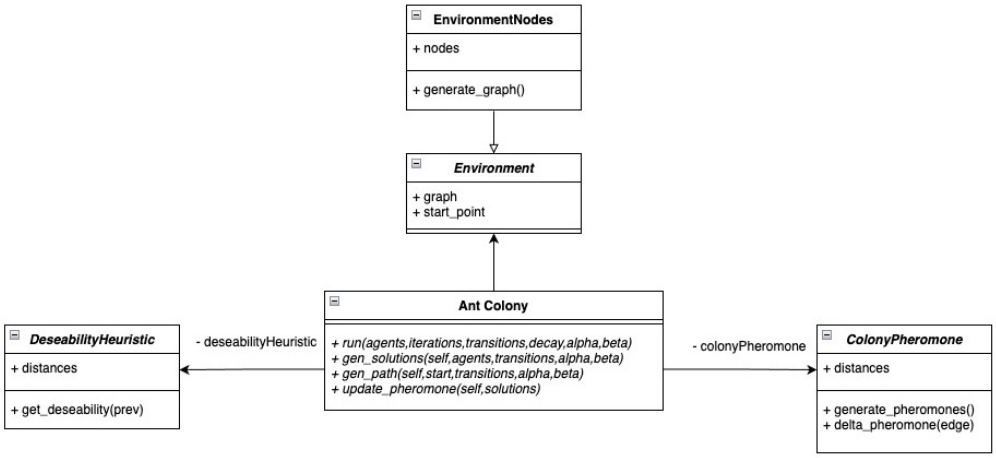


Figura 4.4: Diseño *framework* algoritmo de optimización de colonia de hormigas

A continuación se presentan los componentes correspondientes.

- Componente *Environment*: Este componente representa el espacio de búsqueda sobre el cual trabajará el modelo; en otra palabras permite definir el grafo de búsqueda y la forma en la que este se construye.
- Componente *ColonyPheromone*: Este componente define la forma en la que las feromonas presentes en los vértices del grafo son inicializadas y actualizadas.
- Componente *DeseabilityHeuristic*: Este componente define el comportamiento de la heurística de deseabilidad la cual evalúa un arco según el problema.
- Componente *EnvironmentNodes*: Este componente define la lista de nodos con los que cuenta un ambiente y como se construye el grafo de búsqueda en base a estos nodos.
- Componente *AntColony*: Este componente define el modelo al algoritmo de optimización por colonia de hormigas donde se especifica la forma en la que este se ejecuta y como son generadas las posibles soluciones; esto, utilizando los componentes anteriores.

Validación Una vez diseñado y construido el *framework* este se utilizó para resolver un problema conocido como el problema del agente viajero mencionado anteriormente.

Para el desarrollo del modelo fue necesaria la extensión del *framework*, especialmente en los componentes de *ColonyPheromone* y *DeseabilityHeuristic*. Dichos componentes llevan por nombre *DeseabilityHeuristicTSP* y *ColonyPheromoneTSP* . Una vez esto fue construido el modelo, se ejecutó para varios casos de pruebas donde el desempeño del modelo es comparable con el modelo de algoritmos genéticos, alcanzando buenos resultados.

4.4. Evaluación

Una vez los *frameworks* fueron implementados y revisados se diseñaron una serie de laboratorios prácticos con el objetivo de que los estudiantes implementaran zonas del la metaheurística y solucionaran un problema utilizando el *framework*.

Estos laboratorios se evaluaron en un proceso de revisión externa donde dos estudiantes los realizaron los laboratorios propuestos. Los estudiantes, gracias al *framework* lograron cumplir con la mayoría de los objetivos de los laboratorios.

En esta evaluación cualitativa los estudiantes : (i) enfatizaron el valor de los *frameworks* para la construcción y experimentación con modelos de computación bioinspirada; (ii) resaltaron el diseño de los *frameworks* el cual se considera simple y limpio, (iii) reconocieron la ayuda que brinda el material visual y la documentación de los *frameworks*.

Todo lo anterior son elementos que aportan valor al proceso de aprendizaje. a los *frameworks* construidos y ofrecidos por lo que estos son un material valioso para el diseño de aprendizaje y el curso diseñado.

4.5. Conclusiones

De este trabajo se pueden resaltar una serie de conclusiones. La primera es que la idea original era construir un único *framework* general para los tres modelos pero luego se decidió que esta no era la opción mas viable por lo que se opto por un enfoque donde se diseñara un *framework* por cada tendencia. Por otra lado, con los *frameworks* implementados se evidencia la estructura propuesta por los modelos teóricos en un modelo funcional que puede ser extendido de acuerdo al problema lo que aporta flexibilidad así como la facilidad de realizar experimentos.

El diseño simple logrado en los *frameworks* hace que sea una herramienta intuitiva para que los estudiantes puedan implementar modelos de estos algoritmos y de esta forma resolver problemas. Además, los servicios que ofrece permiten que los estudiantes experimenten con ellos para lograr aprender el funcionamiento y comportamiento de este tipo de modelos junto con sus meta-heuristicas e inspiración biológica.

Finalmente, los modelos resultantes fueron capaces de dar buenas soluciones a problemas de optimización, combinatoria y clasificación.

Parte II

Texto guía

Capítulo 1

Introducción

Desde un principio el hombre ha observado la naturaleza buscando comprender e imitar imitar comportamientos inteligentes observados en fenómenos naturales. Estos comportamientos son dignos de imitar puesto que son soluciones a problemas presentes en la naturaleza que se han perfeccionado durante los años. La naturaleza ha servido de inspiración para resolver problemas en diversas áreas de la ingeniería hasta la actualidad donde se diseñan modelos computacionales heurísticos aplicados a problemas de búsqueda, optimización, reconocimiento de patrones, etc.

La computación natural es el campo de investigación cuyo propósito es comprender los comportamientos que tienen lugar en la naturaleza y de esta manera desarrollar artefactos computacionales bioinspirados. La computación inspirada por la naturaleza es el principal enfoque de este campo de investigación aquí se estudian y analizan los comportamientos de los distintos agentes en la naturaleza, desde los animales hasta las células, esto para desarrollar técnicas de solución de problemas de alta complejidad. [28]

Un aspecto importante a destacar de estos tipos de modelos y algoritmos bioinspirados es que no son explícitamente algorítmicos, donde las soluciones son determinísticas sin tolerancia al error (*hard computing*); en cambio, el funcionamiento de estos modelos consiste en ejecutar ciertos procedimientos que iterativamente van cambiando el estado del modelo y sus propiedades (*soft computing*) generando soluciones no determinísticas. Estos modelos son capaces de realizar tareas de predicción, optimización y búsqueda y normalmente se usan para resolver problemas donde no se conocen soluciones exactas como son los problemas *np-hard*.

El campo de la computación natural se caracteriza por ser interdisciplinario, esto al tener inspiración directa de la biología done el aprendizaje y los conocimientos que aporta esta área permite crear soluciones en un amplio espectro de temas lo que es muy importante para cualquier estudiante de ingeniería. Al conocer esta área se adquieren un conjunto de herramientas importantes para solucionar problemas complejos presentes en distintas disciplinas; además, permite desarrollar un pensamiento interdisciplinario tanto en la creación de soluciones de software como en la resolución de problemas en general, siendo esto una competencia valiosa en la actualidad.

Este libro aborda una introducción a este campo presentando los fundamentos biológicos y la teoría informática de los principales enfoques; dentro de estos enfoques se encuentran las redes neuronales, los algoritmos genéticos y las colonias de hormigas. Estos enfoques o algoritmos bioinspirados le aportan una naturalidad a estas soluciones las cuales se mejoraran mientras se siga investigando esta área. Finalmente es importante resaltar la importancia de esta área ya que los diferentes modelos bioinspirados han demostrado ser eficaces frente a distintos problemas en la ingeniería, donde si se aplicaran métodos o modelos tradicionales seria inviable resolverlos.

Capítulo 2

Computación neuronal. Perceptrón multicapa

2.1. Introducción

Sabemos que el cerebro es capaz de realizar múltiples tareas gracias a la actividad de billones de células nerviosas conocidas como neuronas, cada una de estas células actúa como una unidad individual de procesamiento. Por otra parte estas células se conectan mediante un proceso conocido como sinapsis el cual da origen a una estructura llamado como red neuronal.

El cerebro se se considera al cerebro como el sistema más interesante que conocemos, dicho sistema es complejo, no lineal y computacionalmente paralelo. Este sistema u órgano ha motivado a una gran cantidad de investigadores a lo largo del tiempo, lo que ha generado una gran cantidad de estudios e investigaciones, pero a pesar de todos estos avances una parte de su funcionamiento sigue siendo desconocida.

Dado que el cerebro es el sistema de procesamiento de información más potente conocido es deseable construir modelos informáticos inspirados en este y de esta manera resolver problemas que para los humanos resultan bastante sencillos. Ejemplos de estos problemas pueden ser el uso del lenguaje, la memoria, la resolución de problemas, reconocimiento de patrones, etc. Es con esta motivación que surge el desarrollo de las redes neuronales artificiales hace 60 años donde el principal objetivo era entender y emular los comportamientos observados en el cerebro.

Estas redes neuronales artificiales están diseñadas a partir de elementos observados biológicamente, construyendo de esta manera un modelo donde sus partes e interrelaciones se comportan de forma similar a lo observado en el cerebro: simulando el funcionamiento de las neurona y de las sinapsis para poder desarrollar redes neuronales artificiales con una gran cantidad de unidades de procesamiento.

Recientemente este campo ha resurgido con más fuerza debido al rápido avance en el poder de computo lo que permite el desarrollo de técnicas y arquitecturas más sofisticadas las cuales, junto con el hardware especializado que se ha diseñado y la cada vez mayor cantidad de datos presentes, hacen posible el entrenamiento de modelos que eran imposibles hace unos años, siendo estos capaces de realizar tareas casi con el mismo rendimiento que un humano.[39].

Específicamente este capítulo explora el campo de la computación natural conocido como neurocomputación. Este campo busca inspiración en los procesos biológicos que ocurren en el sistema nervioso para crear modelos computacionales que resuelvan problemas. En este capitulo se detallaran los fundamentos tanto biológicos como informáticos de las redes neuronales, así como su funcionamiento.

2.2. Fundamentos biológicos

2.2.1. La neurona

La neurona es una célula nerviosa presente en el cerebro, pero es diferente a la mayoría de las células que conocemos: no tienen un mecanismo de reproducción y tienen la capacidad de enviar y procesar información done la información aquí está representada mediante impulsos eléctricos o reacciones químicas. Las neuronas al no reproducirse son finitas; pero esta limitación es cubierta gracias a la plasticidad que se logra mediante la creación o eliminación de conexiones que antes no existían.

Una neurona tiene tres componentes principales, dendritas(zonas de recepción), un soma (zona de procesamiento), y un axón (zona de transmisión). Una neurona tiene varias dendritas

y cada una de ellas recibe la señal enviada por la neurona a la cual está conectada. El soma, el cuerpo de la neurona, es la parte encargada de realizar la agregación de las señales de entrada recibidas por las dendritas y cuando esta alcanza cierto umbral dispara una señal de salida. El axón es la parte encargada de enviar los impulsos de salida hacia una dendrita perteneciente a otra neurona, formando de esta manera una sinapsis o conexión. En la figura 2.1 podemos observar la representación gráfica de una neurona y sus componentes.

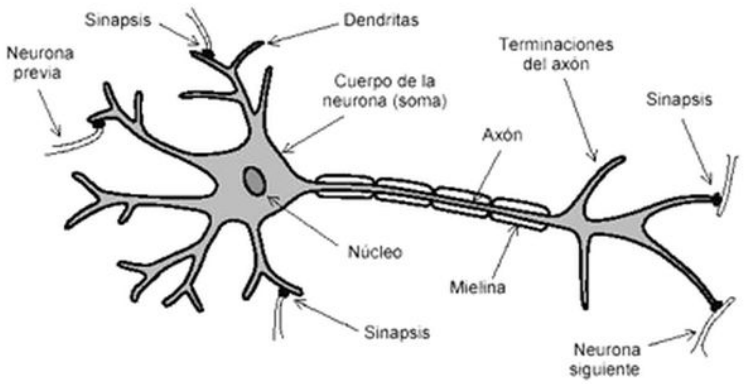


Figura 2.1: Estructura de una neurona [4]

Existen varios tipos de neuronas:

- Sensorial: interpreta los estímulos externos convirtiéndolos en estímulos interpretables por las demás neuronas.
- Motora: realiza la transmisión de estímulos entre el cerebro y los músculos.
- Interneurona: encargada de realizar la comunicación entre los diferentes tipos de neuronas.

2.2.2. Redes

Como se mencionaba anteriormente, una red neuronal se forma mediante las conexiones de varias neuronas a través de la unión de una dendrita de una neurona con el axón de otra neurona, formando circuitos que permiten procesar la información entrante y generar información de respuesta. Una red neuronal está definida por un conjunto de neuronas conectadas mediante sinapsis.

La mayoría de las redes neuronales se forman en el nacimiento o etapas tempranas de la vida, por lo que el desarrollo neurológico es uno de los aspectos mas importantes en las primeras etapas de niñez. Por otra parte cabe resaltar que varias de estas estructuras neuronales también se desarrollan gracias al aprendizaje en donde se generan o pierden conexiones sinápticas entre neuronas.

2.2.3. ¿Cómo funcionan las redes neuronales?

La clave del funcionamiento del sistema cerebral está en las conexiones que realizan las neuronas, es mediante estas que se envían y reciben impulsos, estos son recibidos por las dendritas para luego ser procesados por el soma, una vez la neurona es activada esta envía

una salida mediante el axón, dicha salida se conoce como potencial de acción. Este potencial representa un cambio de polaridad que ocurre cuando la carga en la neurona supera un umbral, es el momento en que se genera un potencial de acción lo que permite el envió a través del axón; mientras dicho umbral no sea sobrepasado no es enviado ningún impulso. Gracias a este proceso es posible el envió de impulsos hacia otras neuronas conectadas a través de la sinapsis, liberando de esta manera neurotransmisores. Las moléculas de los neurotransmisores cruzan la sinapsis y se unen a los receptores de la neurona que los recibe, transmitiendo una señal excitadora o inhibidora.

En la figura 2.2 podemos observar la gráfica correspondiente al potencial de acción.

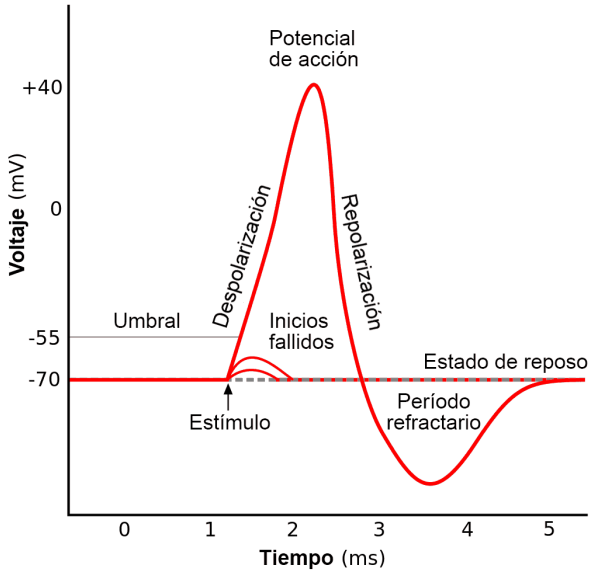


Figura 2.2: Gráfica potencial de acción [58]

Es de esta forma que las redes neuronales trabajan, pero hay otro aspecto a cubrir el cual es el aprendizaje, este ocurre cuando se forman o eliminan sinapsis gracias a una propiedad conocida como plasticidad.

2.2.4. Plasticidad

Los sistemas neuronales se caracterizan por tener cierta plasticidad, esta característica está dada por la capacidad de adaptarse dependiendo del entorno. Esto ocurre de dos maneras, en la creación de nuevas sinapsis y mediante la modificación de sinapsis preexistentes.

Es esta propiedad la que hace posible el aprendizaje ya que al momento de recibir un estimulo externo los receptores del sistema nervioso se encargan de traducir estos estímulos en impulsos eléctricos que las redes neuronales procesan generando salidas las cuales se transforman en acciones físicas. Es en el proceso anterior donde gracias a la experiencia que vamos adquiriendo se modifica la fuerza o pesos de las sinapsis, de esta manera el sistema nervioso se adapta a los nuevos estímulos de un medio ambiente desconocido, aprendiendo en el camino.

2.3. Teoría informática

En esta sección se profundizará en el concepto de redes neuronales artificiales donde se presentará primeramente el modelo mas simple de una red neuronal conocido como perceptrón para luego introducir las redes neuronales multicapa.

Se puede definir a una red neuronal artificial como un modelo computacional compuesto por varias unidades de procesamiento llamadas neuronas, estas unidades se conectan con otras donde la salida de unas se convierten en la entrada de otras, generando de esta forma una salida.

Dado esto una red neuronal artificial tiene la capacidad de resolver varios tipos de problemas, entre estos se encuentran los problemas de clasificación donde el objetivo es predecir la clase correspondiente a una entrada de la red, dicha clasificación puede ser binaria o múltiple. Además se pueden resolver problemas de regresión donde el objetivo es obtener un valor continuo lo mas cercano al valor real.

Este modelo está inspirado directamente de las redes neuronales presentes en el cerebro por lo que hay una serie de conceptos biológicos que son fundamentales para entender el funcionamiento de una red neuronal artificial. A continuación se presentan dichos conceptos claves:

- Las neuronas biológicas reciben señales de entrada mediante las dendritas para posteriormente pasar al cuerpo o soma.
- Todas las señales que llegan al soma se acumulan y de esta manera dan paso a la respuesta de salida la cual se envía gracias a la sinapsis.
- La activación de la neurona ocurre según su umbral de potencial de acción lo que da paso al envió de información a través del axón.
- El axón se conecta con otras dendritas mediante la sinapsis, aquí los neurotransmisores determina que tan fuerte será esta conexión, esto se conoce como peso sináptico.

2.3.1. Perceptron

Arquitectura

El perceptron es el modelo mas simple de una red neuronal con aprendizaje supervisado, este fue propuesto por Rosenblatt [74], este modelo se usa en problemas linealmente separables donde se quiere realizar alguna tarea como clasificación binaria.

En este modelo una neurona cuenta con una serie de entradas o estímulos (dendritas) y una única salida(axón). Cuando ingresan entradas hacia la neurona estas son multiplicadas según el peso sináptico asociado, indicando de esta forma la fuerza de la conexión. Posteriormente estas entradas ponderadas se acumulan en el cuerpo de la neurona donde es procesada generando de esta forma una salida.

Cada ejemplar de entrada de la red está conformado por una serie de valores los cuales representan distintas características. Además, cada uno de estos ejemplares tiene asociado un valor objetivo, este valor debe ser igual que el valor de salida de la red al procesar dicha entrada o ejemplar. Para esto es necesario ajustar los pesos asociados y de esta manera encontrar aquellos pesos los cuales producen las salidas correctas. Esto se conoce como aprendizaje supervisado y se logra utilizando un procedimiento iterativo conocido como algoritmo de aprendizaje.

Primeramente se tiene que la neurona realiza un procedimiento donde dada una entrada de la red se calcula la sumatoria del producto entre los valores de cada unidad de entrada y su peso, sumando al final un valor de sesgo conocido como *bias*. Posteriormente este valor es ingresado en un función de activación obteniendo de esta manera la salida de la red, si dicho valor no corresponde con el valor objetivo la red lo interpretará como un error y realizará una actualización sobre los pesos.

Este proceso se realizará iterativamente hasta que la red no presente errores, obteniendo de esta manera un modelo capaz de realizar tareas de clasificación. Esto es posible siempre y cuando existan un conjunto de pesos que cumplan dicha condición, esto está respaldado por el teorema de convergencia [39].

Como se mencionó anteriormente este modelo está compuesto por un conjunto de unidades de entrada, los pesos asociados a estas unidades y una única neurona donde su valor de salida corresponde al resultado final de la red. Los valores de entrada de la red están dados por $x = x_{i1}, x_{i2}, ..., x_{im}$ donde i indica la neurona o unidad de entrada y m la cantidad de ejemplos que la conforman.

Finalmente en la imagen 2.3 se puede observar gráficamente el modelo de perceptrón descrito anteriormente.

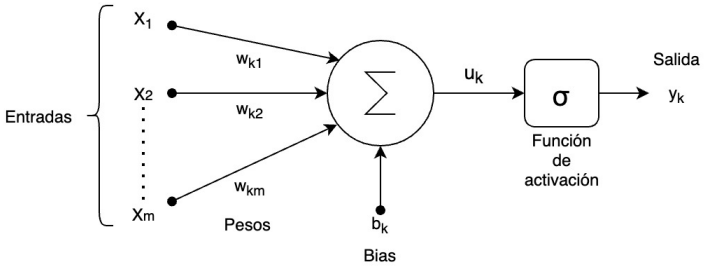


Figura 2.3: Modelo de una neurona

Funciones de activación

En las redes neuronales se usan varias funciones de activación las cuales reciben un valor de entrada y lo transforman generando un valor de salida, estas funciones deben contar con una serie de características para desempeñarse de mejor forma ante distintos problemas, entre estas características se busca que sean funciones no sean lineales, continuas y diferenciables. Las funciones más básicas se detallan a continuación.

1. Umbral: En esta función el valor de salida de una neurona es igual a 1 si la combinación lineal de sus entradas es mayor o igual a 0, sino la salida toma un valor igual a 0. En la figura 2.4 podemos observar la gráfica correspondiente a esta función.

   La definición básica está dada por.

$$y = \begin{Bmatrix} 1 & si & \upsilon_k \geq 0 \\ 0 & si & \upsilon_k < 0 \end{Bmatrix} \tag{2.1}$$

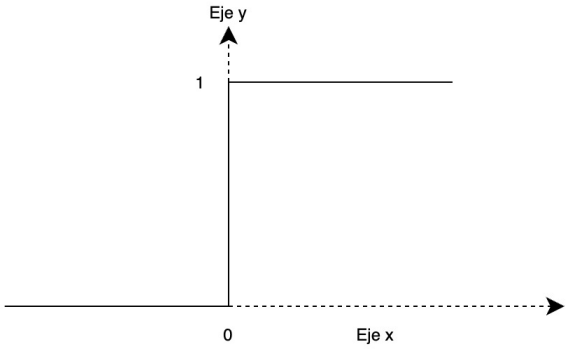


Figura 2.4: Función de umbral

2. Sigmoide: Esta función transforma un valor real de entrada (z) en otro valor en un rango entre 0 y 1, esto teniendo una función cuya curva tiene forma de s. Una característica especial de esta función es que es diferenciable mientras que las funciones del tipo anterior no lo son, esta propiedad es importante al momento de diseñar redes neuronales más sofisticadas. En la figura 2.7 podemos observar la gráfica correspondiente a esta función. La definición de esta función está dada por:

$$y = \frac{1}{1 + e^{-z}} \tag{2.2}$$

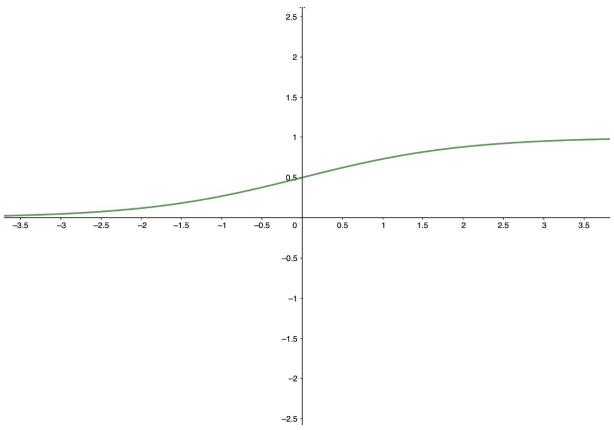


Figura 2.5: Función sigmoide

3. Tangente hiperbólica: Esta función presenta cierta similitud con la función sigmoide pero cuenta con una gran diferencia la cual es que toma valores en un rango entre -1 y 1, dicha característica algunas veces es deseable por lo que esta función es útil en ciertos casos. En la figura 2.6 podemos observar la gráfica correspondiente a esta función.

   La definición de esta función está dada por:

$$y = tanh(z) \tag{2.3}$$

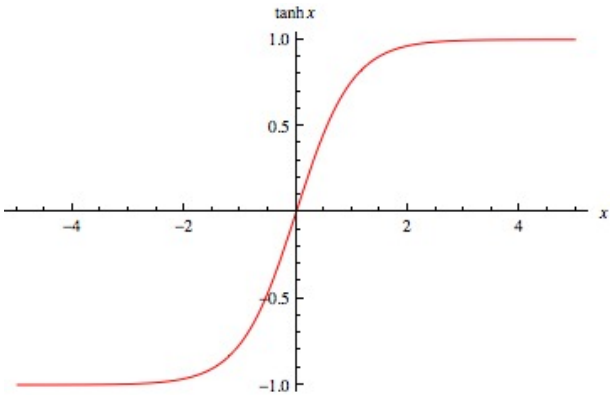


Figura 2.6: Función tangente hiperbólica

4. ReLu: Esta función conocida como rectificador lineal uniforme es una de las mas utilizadas en redes neuronales en la actualidad debido a su simplicidad y fácil computo, además de otras ventajas que ofrece al tener salidas con un valor de cero verdadero. En la figura **??** podemos observar la gráfica correspondiente a esta función.
   La definición de esta función está dada por:

$$y = max(0, z) \tag{2.4}$$

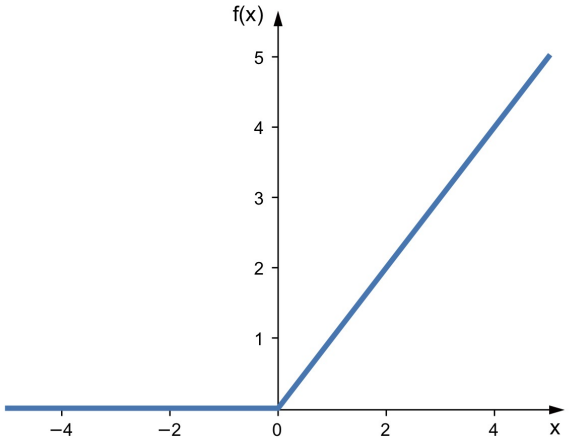


Figura 2.7: Función ReLu

Las funciones de activación mencionadas anteriormente se caracterizan por ser comúnmente usadas para resolver problemas de clasificación pero también se podrían resolver problemas de regresión, la mayor diferencia entre estos dos tipos de problemas es el cambio de la función de activación en la neurona de salida, para solucionar un problema de regresión se debería tener una función de activación lineal. [46]

Problema XOR

El modelo de perceptrón mencionado anteriormente presenta una limitación, esta es la incapacidad de representar funciones no separables linealmente. Esta limitación se demostró en 1969 cuando se evidenció la incapacidad de representar la función xor. Una representación gráfica de este problema se puede observar en la figura 2.8. Allí se ven tres funciones donde es evidente que es posible separar linealmente las funciones AND y OR y es imposible separar el espacio correspondiente a la función XOR.

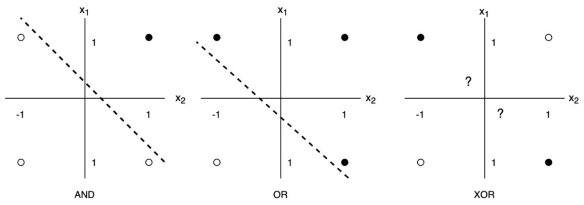


Figura 2.8: Gráfica funciones

La tabla de valores de estas funciones las podemos observar a continuación donde se relacionan dos unidades de entrada con una salida deseada.[10]

x_0	x_1	AND	OR	XOR
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

Dado este limitante fue necesario extender el modelo de perceptrón dando paso a las redes neuronales multicapa o *multilayer perceptron*.

2.3.2. Redes neuronales multicapa

Una red neuronal artificial consiste en un conjunto de múltiples unidades de procesamiento llamadas neuronas. Cada neurona se conecta con otras neuronas mediante conexiones ponderadas, dicho de otra manera se conectan mediante los pesos que representan dicha conexión, formando de esta manera la red neuronal. Estas redes neuronales presentan las siguientes propiedades.

- Un conjunto de unidades de procesamiento(neuronas).
- Un conjunto de uniones(sinapsis) en donde cada una está representada por un peso el cual determina la fuerza de dicha unión, cada peso está definido por w_{jk} el cual determina el peso de la señal de salida de la unidad j sobre la unidad k.
- Un valor de salida y_k por cada neurona, también se interpreta como el estado de activación.
- Una función sumatoria la cual agrega todas las señales de entrada ponderadas por sus respectivos pesos, realizando de esta manera una transformación lineal conocida como preactivación.

- Una función de activación σ que determina el valor de activación dada la preactivación correspondiente.
- Una entrada externa a cada neurona conocida como *bias* b_k

Este conjunto de neuronas se agrupan en capas donde la unión de estas conforman la red neuronal. Definiendo la arquitectura de esta se tiene que el número de capas de una red está dado por L y el número de neuronas de cada capa está dado por $n^{[l]}$ con l como el número de la capa que se está describiendo. Contamos con varios tipos de capas, empezando se tiene una capa encargada de recibir los datos de entrada (x) y otra capa que se encarga de proporcionar la salida del modelo (y), las demás capas se denominan capas ocultas, esta estructura se puede observar en la figura 2.9. Por otra parte en este enfoque se suele utilizar como función de activación la función sigmoide la cual no es lineal y es diferenciable, esto le permite a la red aprender características cuyo comportamiento sea no lineal.

Dado esto se tiene que cada neurona cumple un proceso el cual consiste en recibir un conjunto de entradas las cuales son procesadas generando de esta manera los valores de salida correspondientes. Estos valores se convierten en los valores de entrada hacia las neuronas de la capa posterior donde se realiza el mismo procedimiento hasta llegar a la capa de salida y generar la salida de la red. Esto se conoce como propagación hacia adelante.

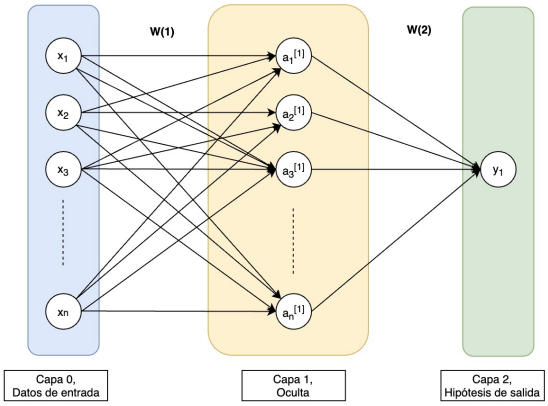


Figura 2.9: Arquitectura MLP

A modo de anotación una red neuronal con una sola capa oculta es capaz de aproximar cualquier función continua, es por esto que se les conoce como aproximadores lineales [25], esto demuestra el enorme potencial que estas tienen para realizar tareas de clasificación y regresión.

Propagación hacia adelante

Como se mencionó anteriormente contamos con una o mas capas ocultas las cuales tienen varias neuronas. En la primera capa un vector de entrada ingresa a la red propagándose hacia las demás capas ocultas, en la primera capa oculta se procesan dichos valores de entrada generando una salida por cada neurona, dichos valores son propagados hacia la capa siguiente donde se realiza el mismo proceso hasta alcanzar la capa de salida, obteniendo de esta manera un resultado final.

El formulación matemática de la propagación hacia adelante está dado por las siguientes fases las cuales se realizan en cada capa.

En la primera fase se calcula la preactivación ($z^{[l]}$), esta es la suma ponderada de las entradas, es decir, la transformación lineal de las entradas dados los pesos sumado a un componente de sesgo.

$$z^{[l]} = W^{[l]} * a^{[l-1]} + b^{[l]} \tag{2.5}$$

Donde $a^{[l-1]}$ son los valores de activación de las neuronas de la capa l-1, $W^{[l]}$ es la matriz de pesos de la capa l y $b^{[l]}$ el vector de sesgos de la capa l.

Posteriormente se calcula la activación de las entradas para cada una de las neuronas de la capa ($a^{[l]}$), esto se realiza mediante una función de activación la cual agrega no linealidad a la red, en este paso la suma ponderada calculada previamente se pasa a la función de activación, dando como resultado el valor de activación de la neurona el cual es propagado como las entradas de la capa posterior.

$$a^{[l]} = g^{[l]}(z^{[l]}) \tag{2.6}$$

Donde $g^{[l]}$ representa la función de activación de la capa l.

Este proceso se realiza por cada una de las capas hasta alcanzar la ultima capa donde la salida de esta está representada por $\hat{Y}$ mientras que Y representa la salida esperada de la red.

Aprendizaje: *Backpropagation*

El algoritmo mediante el cual la red neuronal actualiza sus pesos o en otras palabras aprende se conoce como retro propagación o propagación hacia atrás. El objetivo de este algoritmo es encontrar los pesos que minimicen el valor de una función de error, en otras palabras que las diferencias entre los valores predichos y los valores reales sean mínimas. Esto se logra realizando el calculo del gradiente de la función de error respecto a los pesos de la red.

El nombre de este algoritmo proviene del hecho de que el cálculo del gradiente se propaga hacia atrás a través de la red, calculando primero el gradiente de la capa final y el gradiente de la primera capa de ultimas. Al reutilizar los cálculos parciales del gradiente de una capa para el cálculo del gradiente de la capa anterior se consigue un calculo eficiente. Finalmente, de esta manera las neuronas de la capa oculta reciben una fracción del total del error, basándose en que tanto aportó cada neurona a la salida. Repitiendo este proceso todas las neuronas de la red reciben un gradiente que describe su contribución respecto al error total donde los pesos luego son actualizados según este gradiente.

Las funciones de error calculan el error presentado entre la salida de la red y los valores objetivos. El objetivo de esta función es evaluar el rendimiento del modelo penalizando los malos resultados del modelo. Dado que se busca medir que tan malas son las predicciones, una predicción perfecta equivale a 0 de error, mientras que una mala predicción tiene valores altos.

Funciones de error

1. Error cuadrático medio.

Esta función de error penaliza las diferencias entre los valores esperados y los valores obtenidos por la red, donde entre mayor sea la diferencia mayor es el error, por esto mismo esta función es la mas usada es problemas de regresión.

$$E = \frac{1}{m}\sum_{i=1}^{m}(Y - \hat{Y})^2 \tag{2.7}$$

2. Entropía cruzada.
   Esta función es utilizada en problemas de clasificación puesto que otorga mayores penalizaciones a las predicciones que se alejan de su etiqueta, a pesar que la diferencia absoluta entre los valores sea pequeña.

$$E = -\frac{1}{m}\sum(Y * log(\hat{Y}) + (1 - Y) * log(1 - \hat{Y})) \tag{2.8}$$

Como se mencionó anteriormente en las formulas anteriores se tiene que $\hat{Y}$ es la matriz con los datos de salida predichos de los m ejemplos y Y es la matriz con los datos de salida reales.

Gradiente descendiente Es un algoritmo de optimización utilizado para encontrar los pesos adecuados en un modelo de red neuronal, el objetivo de este algoritmo es encontrar los pesos del modelo que minimizan la función de error. Para ello, realiza cambios en los pesos del modelo, moviéndolos iterativamente a lo largo de un gradiente o pendiente hacia un valor de error mínimo. Esto le da al algoritmo su nombre de "descenso de gradiente".

Como se mencionó anteriormente mediante la propagación hacia adelante se obtiene la salida de la red y es en base a este resultado que se calcula el error de cada neurona de salida. Luego mediante la propagación hacia atrás el gradiente se propaga hacia las capas de atrás y en base a este valor sus pesos se actualizan, de tal manera que el error disminuye, este proceso de propagación hacia adelante y hacia atrás se realiza iterativamente con el objetivo de minimizar la función de error cada vez mas hasta alcanzar valores óptimos y estables.

Detallando el algoritmo de descenso del gradiente este inicia realizando un calculo en la ultima capa, aquí se calcula un termino delta el cual está dado por la diferencia entre la salida esperada y la salida obtenida por la red, donde esta es multiplicada por la derivada de la función de activación. De esta forma tenemos.

$$\delta^{[l]} = (y - \hat{y}){g^{[l]}}'(z^{[l]}) \tag{2.9}$$

Donde y y $\hat{y}$ corresponde a la salida esperada y la salida predicha y ${g^{[l]}}'(z_l)$ es la derivada de la función de activación respecto al valor de preactivación. Este termino representa el cambio de la función de error respecto a los pesos, lo que nos permite realizar la actualización de los pesos moviéndonos hacia un mínimo de la función de error. En el caso de las capas ocultas tenemos que el error que se produce está en función del error producido por las neuronas de la capa posterior, dado esto se tiene que.

$$\delta^{[l]} = (\delta^{[l+1]} * w^{[l]}){g^{[l]}}'(z^{[l]}) \tag{2.10}$$

Aquí tenemos que $\delta^{[l]}$ corresponde al gradiente de la neuronas de la capa l, de esta forma se observa que el gradiente de una capa siempre y cuando no sea la ultima capa depende del

calculo del gradiente de la capa posterior ya que la capa anterior influye directamente en el error de la capa posterior, dado esto el calculo del gradiente es propagado hacia atrás.

Dado esto se realiza la actualización de los pesos donde estos van actualizándose iterativamente a lo largo de un gradiente o pendiente hacia un valor de error mínimo. Esta actualización se realiza de la siguiente manera.

$$W^{[l]} = W^{[l]} - \eta * a^{[l]} * \delta^{[l]} \tag{2.11}$$

Donde η es el hiper-parámetro que regula la tasa de aprendizaje, $a^{[l]}$ es el valor de activación de las neuronas en la capa correspondiente y $\delta^{[l]}$ corresponde al gradiente. Mediante este enfoque los pesos son actualizados de tal manera que el error generado por la red sea lo menor posible, realizando de esta forma el aprendizaje.[10][9]

2.3.3. Algoritmos

Finalmente a continuación se pueden observar los algoritmos correspondiente al modelo de una red neuronal multicapa.

Algoritmo 1: Forward propagation

Data: Input X
Result: activation value
for $l; l \leftarrow 1$ **to** $(numLayers)$ **do**
    $z^l \leftarrow \mathrm{W}^{[l]} * a^{[l-1]} + b^{[l]}$;
    $a^l \leftarrow \mathrm{g}^l(z^l)$

Algoritmo 2: Backward propagation

Data: error
Result: gradients
$\delta^{[numLayers]} \leftarrow$(y-)$\mathrm{g}^{[l]\prime}(z^{[l]})$;
for $l; l \leftarrow (numLayers - 1)$ *downto 1* **do**
    $\delta^{[l]} \leftarrow (\delta^{[l+1]} * w^{[l]}) g^{[l]\prime}(z^{[l]})$;

Algoritmo 3: train MLP

Data: dataset
Result: trained parameters
Inicializar los pesos y el bias;
for $i; i \leftarrow 1$ **to** $(numEpochs)$ **do**
    do forward propagation;
    get loss;
    $\delta^{[l]} \leftarrow dobackwardpropagation$
    for $l; l \leftarrow 1$ **to** $numLayers$ **do**
        $W^{[l]} = W^{[l]} - \eta * a^{[l]} * \delta^{[l]}$

2.3.4. Metodología de desarrollo

Para diseñar un modelo de una red neuronal es necesario seguir una metodología la cual se detalla a continuación.

Definición del problema

El objetivo de esta etapa es especificar de forma clara el problema sobre el que se trabajará y el objetivo que quiere ser alcanzado. En la definición del problema es importante

lograr clasificarlo como un problema de regresión o clasificación. El objetivo lo especificamos seleccionando una métrica de evaluación y definiendo el rango de valor que queremos lograr. Los métricas posibles dependen del tipo de problema como se presenta a continuación:

- Regresión:
  - Métrica de evaluación: Varianza explicada
- Clasificación:
  - Métrica de evaluación: Precisión, recuerdo (recall) o el f1-score.

En esta etapa también se consideran los datos con los que se cuentan para desarrollar la red. En ellos es en dónde está el conocimiento que la red debe aprender.

Análisis y pre procesamiento de los datos

Al momento de definir y estudiar el problema se debe también realizar un análisis sobre los datos que se van a usar, este análisis incluye la limpieza y transformación necesaria de estos, por otra parte de acuerdo a este análisis previo se decide si los datos deberían ser normalizados o no.

Diseño de la arquitectura

El objetivo de esta etapa es diseñar la arquitectura del modelo de red. En esta etapa tomamos decisiones importantes que dependen del tipo de problema, principalmente decidimos: (i) la cantidad de capas y neuronas por cada una de estas, (ii) las funciones de activación de las neuronas de cada una de las capas y (iii) la función de costo. A continuación se presentan las decisiones más comunes en cada caso.

- Regresión:
  - Número de neuronas en la capa de entrada: Número de atributos
  - Número de neuronas en la capa de salida: Una neurona
  - Función de activación (última capa): Lineal
  - Función de costo: Error cuadrático medio
- Clasificación:
  - Número de neuronas en la capa de entrada: Número de atributos
  - Número de neuronas en la capa de salida: Número de clases
  - Función de activación (última capa): Sigmoide [1]
  - Función de costo: Entropía cruzada

El número de neuronas en las capas ocultas generalmente es una potencia de dos debido a como funciona el computo a bajo nivel, este numero estaría entre la cantidad de neuronas de entrada y de salida, disminuyendo a medida que se añaden mas capas en caso de ser necesario. Por otro lado la función de activación de las capas ocultas no depende explícitamente del tipo de problema.

Entrenamiento y sintonización

El objetivo de esta etapa es lograr que la red aprenda los parámetros y sintonizar los hiper parámetros como la tasa de aprendizaje y épocas. Si en esta etapa se evidencia fallas en el

[1] En el caso de problemas de clasificación múltiple es valido usar sigmoide como función de activación de la ultima capa ya que cada neurona actúa como un clasificador binario de la clase

Biologico	Informatico
Estimulo	Entrada de la red
Receptor	Capa de entrada
Neurona	Unidad de procesamiento que recibe entradas de otras neuronas
Pesos sinápticos	Valores de los pesos que determinan el valor de pre-activación de una neurona y conectan la neurona con otra
Cuerpo de la neurona(soma)	Función de activación
Impulso eléctrico	Salida de una neurona propagada hacia las demás
Red neuronal	Conjunto de varias neuronas estructuradas en capas

Cuadro 2.1: Conceptos computación neuronal

rendimiento y desempeño de la red lo ideal sería cambiar la arquitectura y volver a realizar este proceso hasta lograr unos resultados aceptables según los objetivos y métricas establecidas.

En este proceso de entrenamiento se utiliza un subconjunto de los datos totales, este subconjunto generalmente se conoce como datos de entrenamiento y está compuesto por un 80 % de los datos totales, por otra parte también se cuenta con otro subconjunto conocido como datos de prueba cuya finalidad es realizar pruebas sobre el modelo final y está compuesto por el 20 % faltante.

Pruebas

El objetivo de esta etapa es evidenciar la capacidad del modelo sobre datos nuevos. Las pruebas se realizan usando los datos de prueba y considerando las métricas definida. Si se llegan a presentar valores demasiado bajos se debería volver a diseñar y entrenar la red neuronal.

2.4. Aspectos finales

2.4.1. De natural a artificial

En la tabla 2.1 se muestran diferentes conceptos presentes en la teoría biológica y su homologo correspondiente según la teoría informática.

Realizando la comparación entre lo biológico y lo informático, generalmente las neuronas son mucho más lentas en comparación con las compuertas lógicas de silicio: una acción en una compuerta ocurre en el orden de nanosegundos mientras en una neurona ocurren en el orden de milisegundos. Por otra parte, la cantidad de unidades de procesamiento en el cerebro es mucho mayor que en los computadores actuales, ya que se estima que hay aproximadamente 10 billones de estas en el córtex del cerebro humano junto con 60 trillones de sinapsis o conexiones. En comparación las redes neuronales mas sofisticadas actualmente tienen aproximadamente un millón de neuronas.[46]

2.4.2. Línea de tiempo

En la figura 2.10 se puede observar una linea de tiempo con los hitos más importantes del área.

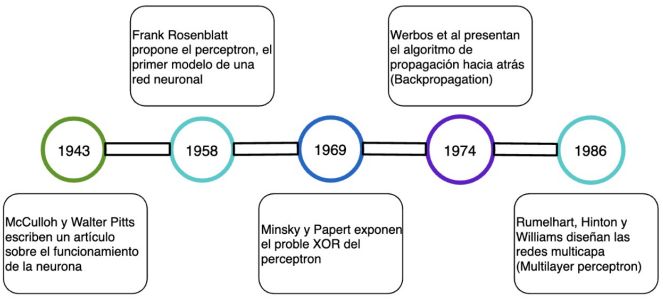


Figura 2.10: Linea de tiempo redes neuronales

Capítulo 3

Computación evolutiva. Algoritmos genéticos

3.1. Introducción

A lo largo de la historia de nuestro planeta se ha observado la enorme capacidad que tienen los seres vivos para adaptarte al cambio presente en el entorno donde habitan. Se han observado organismos que han desarrollado características especiales según las condiciones del entorno, donde pueden haber condiciones extremas, escasez de recursos, etc, de esta manera se han adaptado y logrado sobrevivir. Esta capacidad de evolución y adaptación ha llamado la atención de investigadores de diferentes áreas donde en la biología se busca comprender los fenómenos y en la informática imitar estos comportamientos biológicos para la resolución de problemas.

El constante cambio en el entorno y ambiente es algo que resulta en la muerte de los individuos menos adaptados a su entorno, así como en la supervivencia de los mejor adaptados, este comportamiento conocido como selección natural impulsa la diversidad biológica, comportamiento que se repite a lo largo de las generaciones. Dicha diversidad se refuerza aún más con la aparición de mutaciones en los organismos que pueden tener resultados positivos en la adaptabilidad del individuo, esta diversidad biológica involucra cambios en los organismos que son heredados hacia las futuras generaciones mediante la reproducción, de esta manera se está garantizando una mejor adaptabilidad y supervivencia de la especie ya que las futuras generaciones heredarán las características de organismos aptos, logrando de esta manera la supervivencia de la especie en un entorno coexistiendo con otras especies. Este proceso de adaptabilidad y cambio de los individuos a lo largo de las generaciones se conoce como evolución.

Este comportamiento observado donde los individuos mejor adaptados tienen mas posibilidades de sobrevivir y llegar a evolucionar ha inspirado un tipo de algoritmos conocidos como algoritmos evolutivos. Estos algoritmos son capaces de dar solución a problemas representando posibles soluciones como individuos e imitando el proceso de evolución esperando que luego de un número de generaciones aparezcan individuos que representen soluciones óptimas.

Este capítulo explora el campo de la computación natural conocido como computación evolutiva mostrando los fundamentos biológicos e informáticos, explorando las distintas teorías y detallando los algoritmo genéticos.

La computación evolutiva estudia los modelos informáticos cuya inspiración proviene del comportamiento presente en la teoría evolutiva y genética, principalmente de las ideas de selección natural y diversidad genética. Dentro de esta área contamos con algoritmos genéticos, algoritmos de estrategia evolutiva, programación evolutiva, etc.

3.2. Fundamentos biológicos

3.2.1. Teoría evolutiva

La evolución biológica es la ciencia interesada en el estudio de los cambios y similitudes entre especies y organismos, la diversidad en la naturaleza, y las propiedades de adaptación presentes en esta. La palabra evolución proviene del latín *evolvere* que significa desdoblar, se entiende esto como un sinónimo de cambio, se puede definir la evolución como el proceso en el que ocurre un cambio entre las distintas generaciones en una población. Siendo la teoría evolutiva aquella que estudia estos cambios y comportamientos.

Los primeros estudios en este campo empezaron a principios del siglo XVIII cuando Lamarck propone que cada especie se originó individualmente por generación espontánea, gracias a la existencia de un "fluido nervioso"que hace posible la adaptación de las especies a su entorno. Según Lamarck entre más se ejercite un órgano este atrae más fluido nervioso, fortaleciéndose cada vez mas y estos cambios o nuevas características serian heredadas a las futuras generaciones mejorándose con el pasar del tiempo. [52]

Más adelante, en el siglo XIX, Darwin propone una nueva teoría donde una de las bases es que las especies se forman libremente en la naturaleza, sugiriendo que las especies más abundantes son aquellas que producen variedades bien definidas en los individuos a lo largo de las generaciones. Darwin propone que reproducirse, cambiar y competir por la supervivencia son los pilares de la evolución y mejora de las especies. Esta competencia se conoce como selección natural, este termino se refiere a como prosperan aquellos individuos que presentan ciertas variaciones que facilitan su supervivencia, estas variaciones o diferencias se acumulan durante muchas generaciones sucesivas dando lugar a la aparición de especies nuevas y más adaptadas a su entorno. [28]

En resumen Lamarck propuso una teoría donde el cambio de los individuos está guiado según el ambiente donde dichos cambios son heredados, mientras que Darwin extiende esta idea proponiendo que una población de individuos capaces de reproducirse y sometidos a variación, seguida de selección donde los menos adaptados mueren mientras que los mejores adaptados prosperan y se reproducen, resultado de esto se tienen nuevas poblaciones de individuos cada vez más adaptados a su entorno, este proceso hace que con el paso de las generaciones los individuos sean mejores y resuelvan problemas de manera satisfactoria en su entorno. En la imagen 3.1 se observa una comparación gráfica de las dos teorías donde se puede ver como según Lamarck el cuello de las jirafas va estirándose con el tiempo a medida que esta parte del cuerpo es mas ejercitada, mientras que Darwin propone que los individuos menos adaptados, es decir con el cuello mas corto fallecen mientras que aquellos con un cuello mas largo prosperan dando como resultado generaciones con estas características o mejores gracias a la mutación, siendo finalmente la teoría propuesta por Darwin aquella que es valida por la ciencia. Por otra parte a continuación se resaltan ciertos elementos claves que definen la teoría evolutiva:

- Población
- Variación
- Herencia
- Selección

Estos mismo elementos son los que definen los algoritmos evolutivos presentados en el siguiente capitulo.

Finalmente surgió una teoría genética gracias al trabajo de Mendel, estos principios genéticos formulados complementan la teoría darwiniana al ampliar los conceptos de variación (mutación) y herencia genética.

3.2.2. Teoría genética

Los fundamentos de esta teoría fueron formulados por Gregor Mendel, gracias a su trabajo realizado en una serie de experimentos donde se usaron guisantes [59], en este Mendel seleccionó una serie de cepas de guisantes que se diferenciaban en ciertas características,a dichas características al ser observables y medibles se les conoce como fenotipos, ejemplos de estos

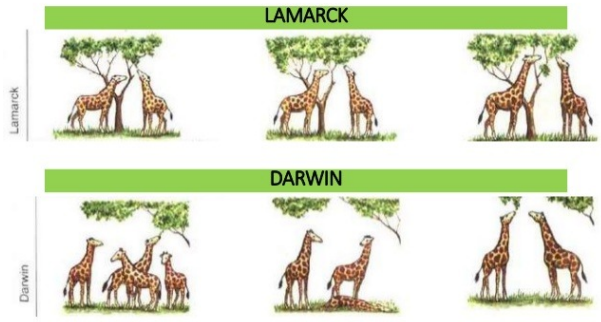


Figura 3.1: Teorías evolutivas

atributos pueden ser el color, textura, etc. Luego de esta separación e identificación Mendel realizó una serie de cruces entre estas plantas dando como resultado nuevas plantas de las que después de analizar e interpretar se concluyó que las características fenotípicas dependían de una serie de factores conocidos posteriormente como "genes", dicha composición genética de un individuo o célula se conoce como genotipo. La palabra genética proviene del griego "gen"que significa descendencia

Resultado de este estudio surgieron tres leyes conocidas como las tres leyes de Mendel, siendo estas leyes la base de la genética. Detallando la primera ley esta enuncia que existen dos tipos de genes, genes dominantes y recesivos, entonces en el caso de los experimentos con guisantes al momento de cruzar dos individuos uno con un gen dominante y otro con un gen recesivo, todos los individuos producto del cruce tendrán las características fenotípicas del gen dominante. Luego la segunda ley enuncia que al realizar el cruce de los individuos resultado del experimento anterior estos producirán una descendencia donde los individuos tendrán un 25 % de probabilidades de presentar las características del gen recesivo el cual es guardado. En la figura 3.2 se observa en detalle el comportamiento descrito anteriormente donde los individuos a través de las generaciones siguen las leyes descritas.

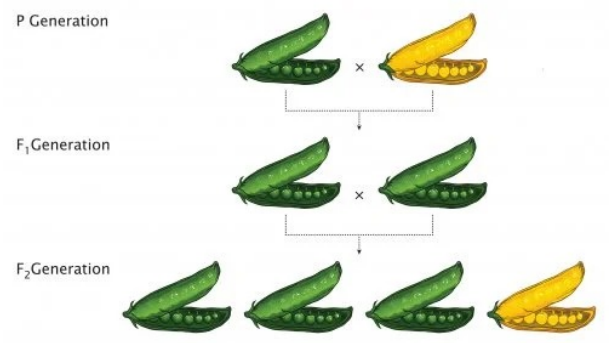


Figura 3.2: Experimento de los guisantes

El material genético de los organismos se encuentra en el núcleo de las células en una estructura conocida como cromosoma. Un cromosoma está compuesto por una molécula de Ácido Desoxirribonucleico (ADN) que contiene los genes, segmentos de ADN. Los posibles valores que puede tomar un gen se conoce como alelos. En la imagen 3.3 podemos observar la

composición desde la célula hasta el ADN.

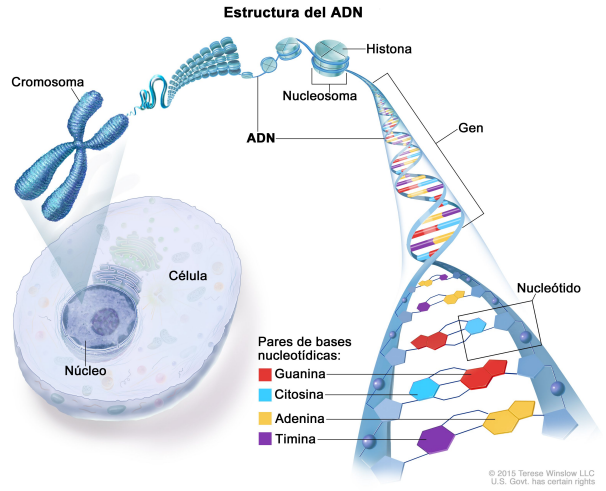


Figura 3.3: Composición cromosomas [30]

El ADN es un polímero formado por una cadena de nucleótidos, donde sus elementos principales son la adenina(A), guanina(G), citosina(C) y timina(T), estos forman puentes de hidrógeno formando de esta forma una estructura tridimensional en forma de doble hélice. Dicha estructura le permite al ADN tener la capacidad de almacenar cantidades abrumadoras de información genética.

Otra característica importante del ADN es su capacidad de replicarse, en la reproducción las cadenas de doble hélice se desdoblan y se forma una nueva cadena que se une con la cadena separada, de esta manera al final se obtienen dos copias idénticas al segmento original, este proceso permite la replicación celular siendo esto la base de la reproducción sexual(meiosis) y asexual(mitosis).

En la reproducción asexual, una célula duplica su material genético y genera dos células hijas, cada una estas contiene una copia exacta del material genético de la célula padre; sin embargo, esto ultimo no está garantizado ya que pueden ocurrir ciertos cambios en la cadena de ADN en este proceso. En la imagen 3.4 podemos observar este proceso.

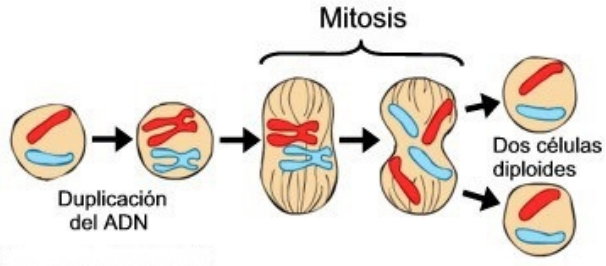


Figura 3.4: Mitosis [1]

La reproducción sexual, por otra parte, se caracteriza por producir células hijas con un material genético diferente a los padres, esto gracias a la recombinación de genes. Específicamente el mecanismo base para la reproducción sexual se conoce como meiosis, este consiste

en la división de una célula diploide (con dos cromosomas), donde el material genético es replicado y luego de dos fases de división celular se producen 4 células haploides (un únicos cromosoma). En la imagen 3.5 podemos observar este proceso: (i) el la interfase, los cromosomas de los padres se replican para luego separarse; (ii) en meiosis i, las partes de cada padre se juntan para crear una nueva descendencia; (iii) en meiosis ii, se generan las cuatro células hijas. Este proceso implica recombinación genética ya que se generan combinaciones de genes en la descendencia que son distintos a los genes de los padres, esta recombinación genética se logra mediante un mecanismo conocido como cruce donde se intercambian ciertos genes dando como resultado nuevos individuos o cromosomas. De esta manera al final tenemos 4 células con un único cromosoma.

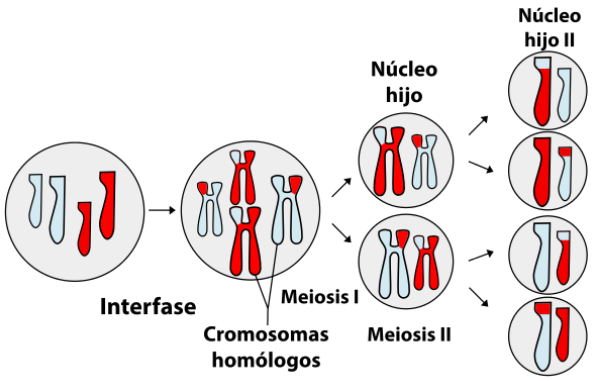


Figura 3.5: Meiosis [2]

Por otra parte existe la posibilidad de que los cromosomas presenten un cambio independiente al cruce, dicho cambio también puede aparecer gracias a la mutación. La mutación es un proceso aleatorio donde el material genético cambia, esto puede ocurrir de varias formas, ya sea mediante la sustitución de un nucleótido por otro, el intercambio de pares de nucleótidos de una zona a otra, así como otros mecanismos de mutación. En la imagen 3.6 se observa como ocurre una mutación en una cadena de ADN o gen.

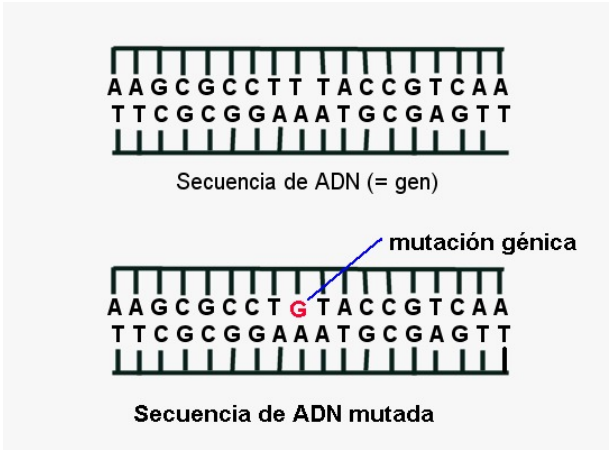


Figura 3.6: Mutación genética [5]

Esta diversificación genética generada por el cruce y la mutación en conjunto con la selección natural son los componentes claves que han permitido la evolución. Siendo la herencia un componente clave al permitir que las características fenotípicas y genotípicas de un individuos sean heredadas a las siguientes generaciones logrando con esto el predomino de los individuos mejor adaptados.

3.3. Teoría informática

En ellos se mantiene un conjunto de entidades que representan posibles soluciones, las cuales se mezclan, y compiten entre sí, de tal manera que las más aptas son capaces de prevalecer a lo largo del tiempo, evolucionando hacia mejores soluciones cada vez.

Un algoritmo evolutivo es aquel que mediante la evolución de agentes que representan posibles soluciones a un problema es capaz de desarrollar y encontrar soluciones óptimas. Esto lo realiza mediante mecanismos o heurísticas inspiradas en la naturaleza, específicamente en la teoría evolutiva y genética. Aplicando esto se tendría un modelo el cual después de varias generaciones o iteraciones se tendría una población o conjunto de soluciones cada vez mejores. Estos tipos de algoritmos tienen una serie de componentes que se enuncian a continuación [16].

- Población: una población está compuesta por un conjunto de individuos (cromosomas) los cuales son la representación de una posible solución y es sobre este conjunto de individuos sobre los cuales el algoritmo evolutivo trabaja.
- Individuo: es un agente que conforma la población. El individuo tiene una serie de valores que representan los genes. Es necesario diseñar una representación para los individuos mediante una estructura de datos, por ejemplo un vector.
- Selección: los individuos se evalúan respecto a la calidad representada mediante la solución, de esta manera los mejores individuos son aquellos que se reproducen formando la siguiente generación.
- Reproducción: los individuos se reproducen mediante el intercambio de genes o valores dentro de su representación, formando de esta manera nuevas soluciones al problema.
- Mutación: es un cambio que sufren uno o mas individuos, dicho cambio se realiza sobre los posibles valores que un gen puede tener, estos valores se conocen como alelos. De esta manera se genera una población con individuos mutados y no mutados.

Resumiendo el funcionamiento de estos algoritmos, se empieza con una población de individuos que representan soluciones, esta población se enfrenta al proceso de selección donde se determina cuales individuos se reproducirán a partir de que tan aptos son, esto mediante una función de aptitud la cual como su nombre lo indica evalúa un individuo según que tan buena es la solución que representa. Posteriormente se crea una nueva generación de individuos mediante la reproducción del grupo seleccionado y finalmente algunos de los individuos generados sufren una mutación. Dicho proceso de selección, reproducción y mutación se repite a lo largo de un número de generaciones, esperando que, en cada una de estas, los individuos representan soluciones cada vez mejores. De esta manera al final del algoritmo la población estará conformada por las mejores soluciones encontradas.

El funcionamiento de estos algoritmos se inspira directamente en los procesos evolutivos y genéticos explicados anteriormente. Al aplicar un conjunto de pasos que simulan el compor-

tamiento natural se logra encontrar o generar una población de individuos que representen soluciones al problema que se intenta resolver.

Este proceso puede verse como un problema de búsqueda en el que a partir de un espacio de posibles soluciones conformado por los individuos de la población se plantea como objetivo encontrar la más óptima.

3.3.1. Algoritmos genéticos

Los algoritmos genéticos son uno de los tipos más populares de algoritmos evolutivos debido a su simpleza y eficacia. Estos algoritmos siguen el mismo esquema básico de los algoritmos evolutivos detallado anteriormente, donde el aspecto que los caracteriza como algoritmos genéticos es el uso de una serie de operadores genéticos los cuales están inspirados en esta teoría.

Individuos

Generalmente los individuos en un algoritmo genético se representan mediante una cadena de valores binarios donde un subconjunto de esta cadena o incluso un valor unitario conforma un gen y toda la cadena generalmente representada en un vector conforma un cromosoma. Cabe resaltar que esta representación binaria se puede extender hacia valores enteros o reales dependiendo de la naturaleza del problema, incluso un cromosoma o individuo puede estar codificado en una matriz y no en un vector exclusivamente. En la Figura 3.7 se observa gráficamente la representación binaria de un individuo.

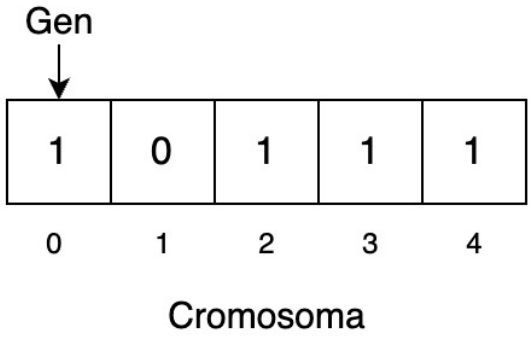


Figura 3.7: Representación del individuo

Para resolver un problema lo primero que se debe hacer es definir la representación del individuo; es decir, determinar la cantidad de cromosomas(tamaño de la población) y la cantidad de genes (longitud del cromosoma). Adicionalmente, se debe seleccionar el conjunto de valores que pueden tomar los genes (alelos).

La inicialización de la población se realiza generalmente de forma aleatoria.

Función de aptitud

Es necesario diseñar una función cuyo objetivo sea calcular que tan bueno es un individuo en el marco del problema, o en otras palabras que tan buena es la solución representada me-

diante el individuo y de esta manera realizar la selección de los mejores individuos.

En los problemas de optimización a veces se conoce de antemano el valor de la función objetivo del individuo que sería una solución en el problema. En este caso, se podría usar como función de aptitud el error cuadrático medio. En la formula siguiente se observa dicha función.

$$f(x) = (g(x) - y)^2 \tag{3.1}$$

donde x representa un individuo y y el valor mas óptimo de la solución, cabe resaltar que no en todos los problemas se conoce previamente el valor óptimo de la solución por lo que es necesario diseñar otro tipo de funciones de aptitud que determinen el desempeño de la solución, un punto clave en ese diseño es que entre peor sea la solución el valor de esta función debe ser menor. [16]

Finalmente cabe resaltar que no siempre se es posible realizar este enfoque con todos los problemas por lo que a veces es necesario definir una función de error que siga una heurística.

A continuación podemos observar el pseudocódigo correspondiente al calculo de la aptitud de la población done la función de aptitud mostrada corresponde al uso de la función de error cuadrático medio.

Algoritmo 4: calcular-aptitud

```
Data: Población(p)
Result: Valor de aptitud (f)
for i; i ← 1 to (longitud(p)) do
    f(p_i) = funcion − aptitud(p_i)
```

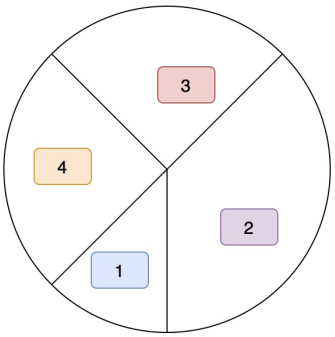
Algoritmo 5: funcion-aptitud (**error cuadrático medio**)

```
Data: individuo(i)
Result: Valor de aptitud
definir función objetivo(g(x)) y valor objetivo (y);
f(i) = (g(i) − y)^2
```

Operador de selección

La selección de los mejores individuos, con base en la función de aptitud, se puede realizar de varias formas; dos de las más conocidas son la ruleta y el torneo.

Ruleta En este enfoque la probabilidad de que un individuo sea seleccionado para reproducirse está directamente relacionada con su aptitud relativa a los otros miembros de la población. Cada individuo representa una porción de la ruleta de forma proporcional a su aptitud y el procedimiento de selección simula una tirada en una ruleta. Esta intuición se puede observar en la figura 3.8.



Cromosoma	Aptitud
Cromosoma 1	12.5
Cromosoma 2	37.5
Cromosoma 3	25
Cromosoma 4	25

Figura 3.8: Estrategia de ruleta

A continuación se presenta la formulación matemática de este mecanismo.

$$p_i = \frac{f(x_i)}{\bar{f}} \tag{3.2}$$

$$\bar{f} = \frac{\sum_{i=1}^{n} f(x_i)}{n} \tag{3.3}$$

Donde p_i es la probabilidad de ser seleccionado del individuo i y $\bar{f}$ es el desempeño medio de la población la cual tiene un tamaño total de n individuos. De esta forma la suma de las probabilidades tendría un total de 1.

Para obtener dicha ruleta se debe realizar el siguiente procedimiento: [35]

1. Definir las secciones correspondientes a cada individuo:
   - $s_0 = 0$
   - $s_1 = p_1$
   - $s_2 = p_1 + p_2$
   - $s_n = p_1 + p_2 + ... + p_n$
2. Generar un número aleatorio r entre el rango [0,1)
3. Seleccionar el cromosoma i tal que $s_{i-1} <= r < s_i$

De esta forma se podría simular el comportamiento de una ruleta donde el proceso de obtener el número aleatorio y el individuo seleccionado se realiza n veces, donde al final se tiene un conjunto de individuos padres con el mismo tamaño que la población inicial.

A continuación se observa el pseudocódigo correspondiente al proceso de selección de la población.

Algoritmo 6: seleccionar-población

```
Data: Población(p)
Result: Población
do calcular-aptitud(población);
do inicializar ruleta;
for i; i ← 1 to (longitud(p)) do
    r = generar numero aleatorio;
    p_i = seleccionar nuevo individuo según ruleta;
```

Cabe resaltar que existen otras formas de realizar esta selección donde un punto a tener en cuenta es la importancia de la diversidad de la población, algunos mecanismos de selección afectan la diversidad al generar conjuntos de individuos con muchas repeticiones entre ellos.

Torneo Frente al problema anterior un mecanismo de selección diferente seria aquel que se haga mediante un torneo. Dicho mecanismo consiste en seleccionar k individuos y realizar un torneo entre estos donde el individuo con mejor aptitud es el ganador y sería añadido al grupo de individuos seleccionados. Este procedimiento se realiza n veces generando de esta manera un nuevo conjunto de individuos. En la imagen 3.9 se observa de forma gráfica este mecanismo.

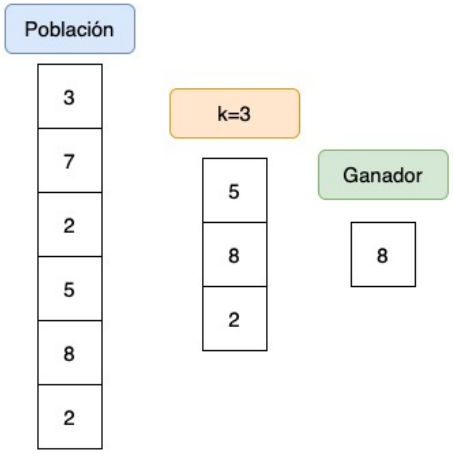


Figura 3.9: Estrategia de torneo

En detalle este mecanismo sigue el siguiente algoritmo:

- Seleccionar k individuos de la población y realizar un torneo entre ellos
- Seleccionar el mejor individuo del subconjunto creado anteriormente
- Repetir los pasos 1 y 2 hasta que tenga el tamaño deseado de población n.

Operador de cruce

Este operador realiza el cruce entre dos individuos padres los cuales son el resultado de aplicar el operador de selección. Este mecanismo permite que los nuevos individuos tengan propiedades de sus padres mediante la herencia de estas mismas y es mediante esto que se obtienen los individuos que conformarán la siguiente generación que podrían llegar a ser mejores que sus padres.

La forma general de realizar este procedimiento es conocida como cruce de un punto simple. En este procedimiento se selecciona una posición de cruce entre los individuos, en base a este los genes a la derecha se intercambian entre si como se observa en la imagen 3.10.

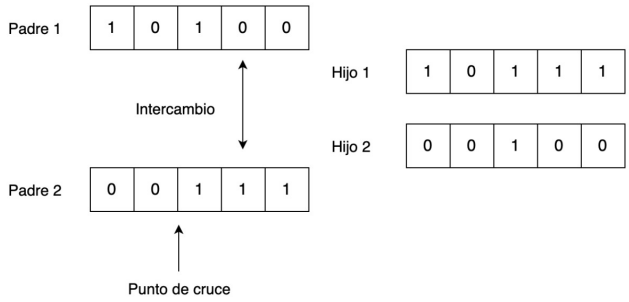


Figura 3.10: Estrategia de cruce

No todos los posibles padres se cruzan. Para cada par de padres seleccionados se genera un número aleatorio entre [0.,1], este número se compara con el parámetro de cruce especificado al inicio del algoritmo, si este es menor se realiza el cruce entre el par de individuos mientras que en el caso contrario simplemente se ignora dicho cruce. [16]

A continuación se observa el pseudocódigo correspondiente al mecanismo de cruce.

Algoritmo 7: cruce

```
   Data: padres,p_c
   Result: p_new población nueva
1  a = generar punto de cruce ;
2  p_new = vector vacío ;
3  for cada par de padres (padres) do
4  |   r = generar numero aleatorio ;
5  |   if r <p_c then
6  |   |   do cruce simple según a ;
7  |   |   añadir nuevos individuos a p_new ;
```

Operador de mutación

Este operador es uno de los más importantes en un algoritmo genético ya que incentiva a la diversidad genética lo que permite descubrir individuos que representen soluciones nuevas y útiles. La forma general de realizar la mutación es recorrer cada uno de los genes de un individuo y generar un número aleatorio donde, según el parámetro de mutación, se realiza el cambio (mutación) o no.

En el caso de que la representación de los individuos sea binaria simplemente se intercambia el valor del gen, es decir si era 1 este pasará a ser 0, este procedimiento se conoce como sustitución puntual[16]. Dicho procedimiento se observa en la figura 3.11.

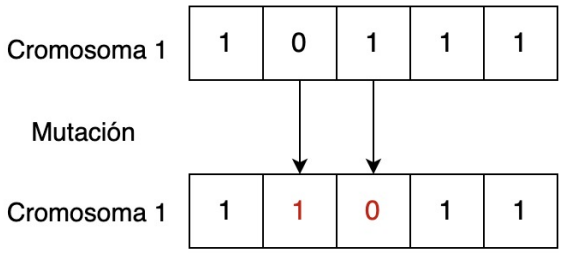


Figura 3.11: Estrategia de mutación

A continuación se observa el pseudocódigo correspondiente al mecanismo de mutación de la población.

Algoritmo 8: mutar-individuos

```
Data: Población(P),p_m
Result: Población
for i; i ← 1 to (longitud(P)) do
    for j; j ← 1 to (longitud(P_i)) do
        r = generar numero aleatorio ;
        if r <p_m then
            do cambio sobre P_{i,j} ;
```

Estrategia de reemplazo

En un algoritmo genético es necesario determinar la forma en la que se formarán las generaciones, es otras palabras decidir la supervivencia de los padres y sus hijos. Para esto tenemos varias estrategias como:

- Remplazo directo, los hijos sustituyen a los padres.
- Remplazo aleatorio, la nueva población se genera de forma aleatoria entre los padres e hijos.
- Remplazo según el peor, los individuos se ordenan según su aptitud y los peores son eliminados hasta tener la nueva generación

En un algoritmo genético clásico generalmente el mecanismo usado es por reemplazo directo, pero dependiendo del problema se podría otro. [16]

Algoritmo completo

Al momento de construir un modelo de un algoritmo genético es necesario inicializar parámetros como la probabilidad de cruce, probabilidad de mutación y el numero máximo de generaciones, de esta forma el algoritmo genético ya estaría listo para funcionar.

Luego junto a la población inicializada se realiza la selección, cruce y mutación de esta por un número determinado de generaciones, obteniendo de esta manera una población óptima para dar solución al problema. A continuación se observa el pseudocódigo que describe un

algoritmo genético.

Algoritmo 9: Algoritmo genético

```
Data: Parámetros: probabilidadDeCruce, probabilidadDeMutación, numeroGeneraciones
Result: Población
población=inicializar población;
p_c = probabilidadDeCruce;
p_m = probabilidadDeMutación;
for i; i ← 1 to (numGeneraciones) do
    población = seleccionar-población(población);
    población = cruce(población,p_c);
    población = reemplazar-población(población);
    población = mutar-individuos(población,p_m);
```

3.3.2. Metodología de desarrollo

Para diseñar un modelo de un algoritmo genético es necesario seguir cierta metodología la cual se detalla a continuación.

Definición del problema En esta etapa se define de forma clara el problema sobre el que se trabajará para luego estudiarlo y obtener una mejor comprensión de este.

Definir la representación del problema En esta etapa se define la forma en la que se van a representar las distintas soluciones es decir, los individuos. Esta etapa es bastante importante ya que aquí se especifica sobre qué estructura va a trabajar todo el algoritmo. Por otra parte, es al momento de definir las distintas restricciones que puede tener los individuos que representan las soluciones al problema y cómo se van a manejar dichos casos.

Definir la función de aptitud En esta etapa se determina la función de aptitud, dicha función tiene como objetivo evaluar la calidad de una solución ya sea mediante heurísticas o comparándola con el valor de la mejor solución conocida en caso de ser posible. Esta función es bastante importante ya que tiene la responsabilidad de determinar cuales son las mejores soluciones lo que impacta totalmente el comportamiento del algoritmo.

Definir los operadores genéticos En esta etapa se definen cuales serían los mejores operadores genéticos para resolver el problema, esto teniendo en cuenta la naturaleza del problema y su representación. Para definir e implementar los operadores de selección, reproducción o cruce y mutación contamos con mecanismos ya definidos como lo son la selección por torneo o el cruce de un solo punto, siendo estos bastantes conocidos, aun así en algunos casos puede ser necesario extenderlos para adaptarlos al problema. Cabe resaltar que no siempre los operadores se comportan como se tiene pensado por lo que luego estos pueden ser cambiados.

Ejecutar y validar el algoritmo Una vez todos los componentes del algoritmo están implementados el modelo de algoritmo genético está listo para ser ejecutado, es esta fase donde se ejecuta el programa y de acuerdo a los resultados se valida si este está funcionando correctamente según lo esperado, es decir que con el paso de las iteraciones la función de aptitud va mejorando o en otras palabras verificar que tenga un comportamiento creciente y estable.

Aquí el algoritmo podría comportarse de forma extraña, esto posiblemente se deba a los hiper-parámetros con los que se cuenta, estos al afectar la forma en la que el algoritmo funciona se debe realizar una experimentación con estos con el objetivo de encontrar aquellos hiper parámetros que mejor soluciones obtienen, dentro de los hiper-parámetros se tienen la tasa de reproducción y mutación. Es de esta manera donde al seguir esta metodología se logra resolver un problema usando un algoritmo genético.

3.4. Aspectos finales

3.4.1. De la natural a lo artificial

En la tabla 3.1 se muestran diferentes conceptos presentes en la teoría biológica y su homologo correspondiente según la teoría informática.

Biológico	Informático
Individuo (1 o más cromosomas)	Representación de un individuo solución. Representa una posible solución al problema
Población	Conjunto de individuos solución
Aptitud	Valor que cuantifica la calidad del individuo solución
Selección	Operador responsable de elegir los individuos solución que se recombinarán
Cruce	Operador mediante el cual dos individuos solución padres se recombinan, generando una nueva solución
Mutación	Operador que realiza cambios aleatorios sobre un individuo solución

Cuadro 3.1: Conceptos computación evolutiva

3.4.2. Línea de tiempo

En la figura 3.12 se puede observar una linea de tiempo con los hitos más importantes del área.

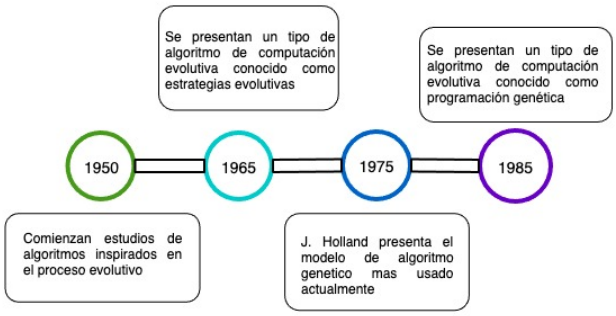


Figura 3.12: Linea de tiempo computación evolutiva

Capítulo 4

Inteligencia de enjambre. Algoritmo de optimización de colonia de hormigas

4.1. Introducción

En la naturaleza existen una gran cantidad de especies que presentan comportamientos colectivos o sociales que ofrecen varios beneficios. Por ejemplo, la convivencia en grupos facilita la recuperación de alimentos, reduce la probabilidad de ataques, permite la división del trabajo, etc. Las especies que presentan dichos comportamientos tienen la capacidad de realizar tareas complejas que serian imposibles de otra forma. Los insectos son los animales que capturan el mayor interés: hormigas, abejas, termitas, etc.

El comportamiento de insectos como las hormigas ha llamado especialmente la atención de diversos investigadores ya que la integración y la coordinación de tareas de los individuos que componen estas colonias no requieren de un jefe central que de órdenes a todos. En cambio, los individuos realizan tareas siguiendo reglas simples, donde la ejecución de estas tareas da origen a un tipo de inteligencia colectiva donde las colonias son capaces de llevar a cabo tareas complejas como encontrar el camino más corto entre dos puntos y construir estructuras complejas.

El origen de esta inteligencia colectiva es uno de los interrogantes que se han planteado a cerca del funcionamiento de la naturaleza, por ejemplo han surgido preguntas sobre ¿Cómo el comportamiento individual de un conjunto de individuos dan como resultado un comportamiento grupal complejo? o ¿Cómo una colonia de hormigas encuentra la ruta óptima entre la colonia y su fuente de alimento?.

Estos comportamientos sociales han inspirado la creación de algoritmos que toman como base los distintos comportamientos de los enjambres para desarrollar algoritmos o herramientas computacionales que hacen uso de los principios biológicos por los que se rige el comportamiento colectivo para resolver problemas cotidianos y de ingeniería relacionados especialmente con la optimización y búsqueda. El campo de la computación natural que estudia estos comportamientos biológicos y lo aplica a la resolución de problemas se llama inteligencia colectiva o de enjambre (*swarm intelligence*). [15] [28]

La inteligencia de enjambre (*swarm intelligence*) corresponde a aquellos algoritmos diseñados para la resolución de problemas los cuales están inspirados en el comportamiento inteligente colectivo de insectos sociales y otras sociedades animales.

Este capítulo explora el campo de la computación natural conocido como inteligencia de enjambre y muestra como los algoritmos bioinspirados en comportamientos de inteligencia colectiva son capaces de solucionar problemas de ingeniería.

4.2. Fundamentos biológicos

En la naturaleza existen ciertos comportamientos que les permite a algunos organismos realizar tareas de forma colectiva las cuales serían imposibles de realizar de forma independiente, construyendo de esta forma sociedades que presentan una inteligencia colectiva. Lo anterior ocurre gracias a interacciones entre los miembros de un conjunto de animales o sociedad; dichas interacciones realizan principalmente gracias a la comunicación química y visual, siendo posible de esta manera una auto organización que genera dicho comportamiento colectivo inteligente.

Un enjambre se puede definir como un conjunto de agentes que interactúan en un ambiente siguiendo unas reglas dadas por este. Ejemplos de enjambres en la naturaleza pueden ser una colonia de hormigas, donde los agentes son hormigas; una bandada de pájaros, donde los

agentes son pájaros; o el tráfico, el cual es un enjambre de automóviles. En este contexto, un agente puede entenderse como una organismo presente en un ambiente o entorno, capaz de sentir y actuar sobre este donde las acciones pueden incluir la modificación del ambiente o interacciones con otros agentes. Es justamente gracias a este comportamiento de los agentes y su interacción con otros agentes que surge una inteligencia colectiva a partir de las acciones individuales de los agentes presentes.

Como se mencionó anteriormente, la colonia de hormigas es uno de los enjambres más interesantes por lo que en este texto se profundizará en el comportamiento de este enjambre.[31] [28]

4.2.1. Colonia de hormigas

Las hormigas son los insectos sociales que más llaman el interés de diversos investigadores siendo estos uno de los insectos más estudiados. Los biólogos están interesados en estudiar cómo estos animales tan simples son capaces de realizar tareas tan complejas como la búsqueda de alimento y la agrupación de cadáveres. Los informáticos buscan replicar en sistemas artificiales los patrones observados en las hormigas para colaborar en la comprensión y usarlos para solucionar problemas. El estudio de estos insectos sociales, aporta a una mejor comprensión del funcionamiento de la naturaleza.

Si observamos una colonia de hormigas nos percatamos de su gran capacidad para trabajar en conjunto y adaptarse al entorno. En las colonias de hormigas no existen individuos con el rol de supervisor o mando central, por lo que los miles de hormigas funcionan sin la dirección de un individuo; lo cual es un aspecto que genera bastante interés en este tipo de sociedades o colonias. Por el contrario, estas colonias funcionan gracias a a la inteligencia colectiva que emerge de la colaboración e interacción entre las hormigas que les permite como colonia resolver problemas complejos mientras que cada hormiga solo sigue reglas simples.

Las colonias de hormigas viven en un ambiente cambiante, al que deben responder, por lo que deben estar preparadas para realizar varias tareas de forma organizada y rápida. La asignación de estas tareas se realiza de forma dinámica sin ninguna jerarquía. A continuación se explica el comportamiento observando en las colonias de hormigas al buscar su alimento, proceso conocido como forrajeo. [28]

Búsqueda de alimento

Las hormigas se comunican entre ellas por medio del tacto y el olfato, ya que estos insectos son casi ciegos, estos individuos perciben el olor de las otras hormigas a su alrededor, específicamente perciben las feromonas segregadas por otros individuos. Una feromona es una sustancia química segregada por las hormigas para comunicarse; de esta manera una hormiga es capaz de percibir a otras y obtener información valiosa como su procedencia o si ha estado afuera o adentro de la colonia. Es gracias a esta sustancia química que las hormigas interactúan unas con otras, siendo posible la realización de varias tareas, una de estas es la identificación y recolección de comida.

En detalle el proceso de formación de caminos puede dividirse en dos pasos: trazado y seguimiento de senderos. El trazado de senderos comienza cuando las hormigas patrulleras salen de la colonia para explorar los alrededores, esto con el objetivo de identificar si el sitio es seguro y libre de riesgos; una vez identificado el lugar como seguro, estas regresan y le indican

a las hormigas recolectoras que es seguro salir, estimulándolas mediante el contacto de sus antenas. [18]

Al principio solo unas pocas hormigas salen en busca de alimento. Cuando una hormiga encuentra una fuente de alimento, regresa al nido y deja un rastro de feromonas detrás suyo, por lo que se realiza una modificación en el entorno, esto hace que otras hormigas sean estimuladas y reclutadas para dirigirse hacia la fuente de alimento encontrada. Las hormigas reclutadas entonces siguen el rastro hasta la fuente de alimento para recolectarlo y llevarlo a la colonia, depositando en el viaje de regreso una carga de feromonas que fortalece el camino o ruta hacia la fuente de alimento formando de esta manera un rastro. El rastro de feromonas señalan la ruta que las otras hormigas seguirán en adelante, explotando de esta manera este camino. Sin embargo, existe la posibilidad de que algunas hormigas se alejen del camino indicado por el rastro, explorando el ambiente y descubriendo nuevas rutas potenciales. Es de esta manera que se va definiendo el camino final que seguirán las hormigas el cual generalmente es el camino más corto. Adicionalmente, las feromonas se evaporan con el tiempo; esto significa que al pasar el tiempo el camino puede perder fuerza si no hay refuerzo de feromonas.

El refuerzo, la aleatoriedad de la exploración y la evaporación son las claves fundamentales para que las hormigas logren encontrar el camino más corto entre la fuente de alimento y la colonia. Esto ocurre porque si el camino que se está explotando actualmente, con mayor nivel de feromona, no es el mejor, un camino más corto puede ser descubierto mediante la exploración de caminos alternos. Dado esto con el tiempo el nuevo camino más corto descubierto presentará un nivel de feromonas superior. [43] [28].

En la figura 4.1 se puede observar el comportamiento descrito anteriormente. Primeramente una hormiga identifica una fuente de alimento y viaja a la colonia dejando en el camino un rastro de feromonas, para luego iniciar fase de exploración y recolección, donde a medida que avanza este las hormigas son capaces de encontrar la ruta mas corta gracias al uso de las feromonas.

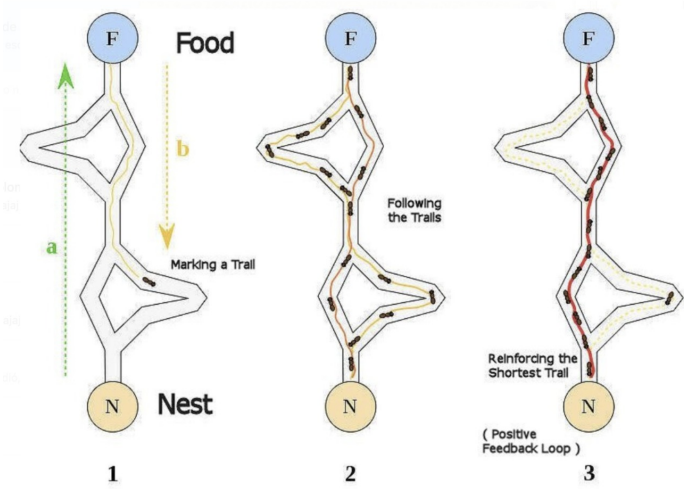


Figura 4.1: Comportamiento de forrajeo [63]

Gracias a este comportamiento individual, que es guiado mediante una serie de reglas simples, que la colonia toma decisiones que en conjunto hace que se comporten como un grupo

auto organizado e inteligente. El patrón de búsqueda de alimento descrito anteriormente brinda la inspiración que da lugar a algoritmos útiles para la resolución de problemas, especialmente problemas de optimización y búsqueda donde el principal algoritmo es la optimización por colonia de hormigas.

4.3. Teoría informática

Un algoritmo de inteligencia colectiva se basa en un conjunto de agentes simples, unas reglas de comunicación local entre estos agentes y un mecanismo de control perteneciente a cada agente. Un agente en el campo de la inteligencia artificial es aquel componente o entidad que presenta la capacidad de percibir su entorno, procesar los datos presentes en dicho entorno y actuar sobre este de acuerdo al resultado del procesamiento de la información percibida y un conjunto de reglas, esto para cumplir una tarea objetivo.

En el caso de la inteligencia colectiva hablamos de agentes ya que los algoritmos presentes en este campo están compuesto por una serie de agentes o individuos. Estas entidades perciben su entorno local, para luego tomar decisiones, dichas decisiones dependen de las acciones y comportamiento de otros agentes que han actuado previamente y pueden modificar el ambiente o interactuar con otros agentes. De esta manera un agente es capaz de representar una hormiga, abeja o incluso una célula.

En los sistemas de inteligencia colectiva contamos con 5 principios [61]:

- Proximidad: los individuos tiene la capacidad de interactuar para formar relaciones sociales.
- Calidad: se deben poder evaluar las interacciones de los agentes con el ambiente y entre sí.
- Diversidad: la diversidad mejora la capacidad de reacción del sistema y aporta cambios en la heurística del sistema, simulando el comportamiento inesperado de la naturaleza.
- Estabilidad: los individuos deben tener un comportamiento estable frente a posibles variaciones.
- Adaptabilidad: Se tiene la capacidad de adaptación frente a cambios en el ambiente.

Dado lo anterior un sistema de inteligencia colectiva o de enjambre es aquel compuesto por un conjunto de individuos capaces de interactuar entre sí y con el entorno. Por otra parte la inteligencia de enjambre es una propiedad emergente del sistema debido a la interacción de sus agentes gracias a los principios de proximidad, calidad, diversidad, estabilidad y adaptabilidad.

4.3.1. Algoritmo de optimización por colonia de hormigas

El algoritmo de optimización de colonia de hormigas (ACO) fue propuesto inicialmente por Marco Dorigo y compañía [34], este entra en la categoría de algoritmos que tienen presentes métodos heurísticos para solucionar problemas, especialmente problemas de optimización combinatoria donde el objetivo es hallar la solución óptima que minimiza o maximiza una función objetivo dada. Esta representación facilita la implementación del algoritmo donde se tienen varios agentes u hormigas que recorren el grafo construyendo posibles soluciones.

El algoritmo de optimización por colonia de hormigas se utiliza para encontrar la ruta más corta en un grafo. Una posible solución está representada por una ruta en el grafo que conecta un nodo inicial con un nodo destino, donde el arco que conecta dos nodos tiene un

costo asociado. El costo de la solución es la suma de los costos de los arcos que la conforman. En la figura 4.2 se observa la representación de un grafo donde los nodos están relacionados con otros mediante pesos en sus arcos.

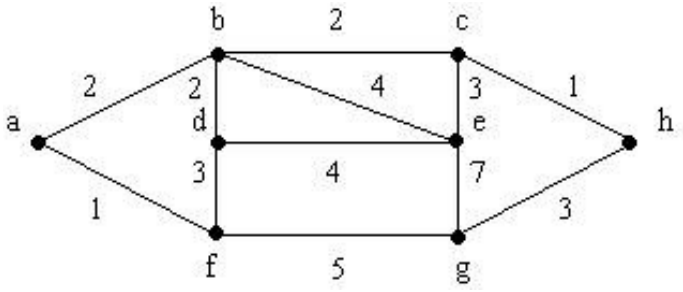


Figura 4.2: Grafo ponderado [3]

Además, asociado con cada arco (i, j) se tiene una variable $\tau_{i,j}$, llamada rastro de feromonas artificiales, que indica que tan bueno es este arco. De esa forma se tiene que cada hormiga artificial deposita un rastro de feromonas en un arco especifico.

En el algoritmo de optimización por colonia de hormigas (ACO) realiza una dos pasos durante un número máximo de iteraciones. Estos pasos son (i) la construcción una solución por cada hormigas y (ii) la actualización del rastro de feromonas presente en los distintos arcos del grafo. De esta forma se tiene que N hormigas recorren el grafo durante un número determinado de iteraciones o hasta que se cumpla alguna condición de parada, en dichas iteraciones cada agente realiza un recorrido a través del grafo construyendo de esta forma una solución. Es en este recorrido donde se depositan las feromonas a cada arco las cuales cada agente utilizará para tomar mejores decisiones en sus recorridos futuros. Gráficamente en la figura 4.3 se puede observar como una solución (camino) está conformada por el camino seguido de una hormiga desde un punto de inicio hasta un punto final.

En el algoritmo de optimización por colonia de hormigas (ACO) las hormigas de la colonia recorren el grafo durante un número determinado de iteraciones o hasta que se cumpla alguna condición de parada. En dichas iteraciones cada agente realiza un recorrido a través del grafo construyendo de esta forma una solución que corresponde al camino seguido de una hormiga desde un punto de inicio hasta un punto final. Al final de la iteración, se depositan las feromonas a cada arco del mejor camino las cuales las guiarán para tomar mejores decisiones en sus recorridos futuros. Gráficamente en la figura 4.3 se puede observar como una solución (camino)

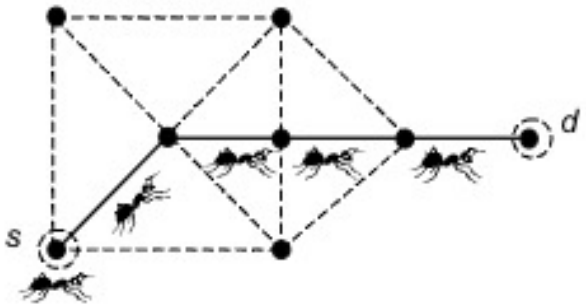


Figura 4.3: Construcción de solución *obtenida de: [28]*

Las hormigas construyen las soluciones de forma probabilística. La probabilidad de que una hormiga se mueva de un nodo i a un nodo j (añadiendo el nuevo nodo como parte de la solución) está en función tanto del rastro de feromonas τ como de una heurística de deseabilidad η.

La deseabilidad heurística depende del problema a resolver, por ejemplo dado el caso en que el objetivo sea encontrar la ruta con el mínimo costo, la deseabilidad se podría determinar calculando el inverso del costo del arco. Por otro lado, la actualización de los rastros de feromonas se realiza en función de la tasa de evaporación ρ y del valor de feromonas en dicho vértice. Las reglas de decisión de la hormiga que recorre el grafo se explicaran a continuación. [28]

El rastro de feromonas τ representa la capacidad de explotación del sistema al tener presente que tan fuerte es dicho rastro en un arco para seleccionarlo; mientras que, la deseabilidad heurística η representa la capacidad de exploración del sistema, ya que expresa, en función de una heurística, que tan deseable para solución del problema es para una hormiga seleccionar ese arco lo que incentiva la elección de caminos donde posiblemente el rastro de feromonas no sea el mas fuerte.

En la siguiente formula se describe la probabilidad que tiene una hormiga de pasar de un nodo a otro.

$$p_{i,j} = \frac{(\tau_{i,j})^{\alpha}(\eta_{i,j})^{\beta}}{\sum_{j \in J_i}(\tau_{i,j})^{\alpha}(\eta_{i,j})^{\beta}} \tag{4.1}$$

Donde J_i es el conjunto de nodos conectados al nodo i, $p_{i,j}$ representa la probabilidad que tiene una hormiga de moverse de un nodo i al nodo j, $(\tau_{i,j})$ es el nivel de feromonas en asignado al arco correspondiente y $(\eta_{i,j})$ es el valor de deseabilidad de dicho vértice, por otra parte α y β son los parámetros que controlan el impacto de estos valores y su influencia en el modelo.

La actualización de los arcos seleccionados se realiza siguiendo la siguiente formula:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \Delta\tau \tag{4.2}$$

Donde τ_{ij} es el nivel de feromonas en dicho arco, $\Delta\tau$ es una constante que indica la cantidad de feromonas a depositar y ρ es el parámetro de evaporación, esto para evitar una rápida convergencia en soluciones subóptimas.[28]

Analizando las formulas anteriores se observa que cuando una hormiga se mueve hacia un nodo j desde un nodo i el nivel de feromonas del arco (i, j) aumenta, por lo que la probabilidad de que este vértice sea seleccionado por otra hormiga aumenta, reforzando de esta manera el arco o vertice del grafo. A continuación se especifica el pseudo código del algoritmo de optimización por colonia de hormigas.

Algoritmo 10: Optimización por colonia de hormigas (ACO)

```
   Data: hiper-parámetros, n_iteraciones, N: número de hormigas
   Result: mejor: mejor solución
 1 Inicializar τ aleatoriamente ;
 2 Asignar N hormigas aleatoriamente en los e nodos ;
 3 Inicializar mejor ;
 4 for t; t ← 1 to (n_iteraciones) do
 5     for i; i ← 1 to (N) do
 6         Ubicar la hormiga i en un nodo inicial del grafo valido;
 7         Construir la solución aplicando la regla de transición probabilística en la ecuación (4.1) ;
 8         if costo de la solución <costo de mejor then
 9             mejor ← solucin
10     Actualizar el rastro de feromonas de la mejor solución según la ecuación (4.2)
```

En el pseudocódigo anterior se describe el algoritmo de optimización por colonia de hormigas, este recibe los distintos hiper-parámetros como la tasa de evaporación, la constante de feromonas y los parámetros de α y β, y retorna *mejor* que es la mejor solución encontrada.

En el algoritmo se observa que cada hormiga construye una solución apoyándose en una regla de transición probabilística, este proceso se realiza durante un número de iteraciones donde al final el algoritmo generalmente es capaz de encontrar buenas soluciones.

A continuación se enuncian las principales características de este algoritmo:

- Las hormigas se usarán para construir las soluciones.
- Una regla de transición probabilística determinar el siguiente arco que conformará la solución de cada hormiga
- El nivel de feromonas de cada arco indica qué tan bueno es este arco considerando las soluciones previas.
- Una heurística de deseabilidad define la probabilidad de que una hormiga seleccione un arco dada una función heurística depende del problema a resolver

4.3.2. Metodología de desarrollo

A continuación se detalla la metodología necesaria ara diseñar un modelo de optimización por colonia de hormigas.

Definición del problema En esta etapa se define de forma clara el problema sobre el que se trabajará, esto con el objetivo de conocer a fondo el problema a tratar, los distintos componentes que lo conforman y que se espera del algoritmo.

Representación del problema Diseñar la estructura sobre la cual el algoritmo trabajará; es decir, el grafo de búsqueda correspondiente al problema. Aquí se deben definir los distintos componentes que conforman el grafo como lo son los nodos iniciales, nodos finales y los pesos de los arcos.

Heurística deseabilidad Definir la función heurística de deseabilidad, está función evalúa que tan bueno es un arco según el problema, donde un valor mayor indica que dicho arco es deseable. Esta función es muy importante para que el algoritmo se comporte adecuadamente aprovechando el conocimiento sobre el problema.

Construcción del algoritmo solución Una vez definidos los componentes anteriores se procede a implementar el algoritmo diseñado donde mediante la definición de los componentes estos se plasman en código siguiendo el diseño del algoritmo de optimización por colonia de hormigas. Una vez está implementado el modelo de (ACO) los distintos agentes generarían soluciones potenciales al recorrer el grafo.

Experimentación e iteración Hallar los valores de los hiper parámetros que dan como resultado las mejores soluciones posibles, dicha búsqueda se realiza de forma iterativa probando distintos valores y de esta manera sintonizar los parámetros para lograr un modelo eficaz y sólido.

Dentro de los hiper parámetros se tienen a *alpha* y *beta*, los cuales afectan la influencia del nivel de feromonas y la heurística de deseabilidad respectivamente. Además se cuenta con otro hiper parámetro ρ que es la constante de evaporación.

4.4. Aspectos finales

4.4.1. De la natural a lo artificial

En la tabla 4.1 se muestran diferentes conceptos presentes en la teoría biológica y su homologo correspondiente según la teoría informática.

Biológico	Informático
Hormiga	Agente (hormiga artificial) utilizado para construir una solución al problema
Colonia de hormigas	Colonia de agentes
Rastro de feromonas	Modificación del entorno realizada por la población de hormigas. Esta permite evaluar la calidad de un arco en el grafo de búsqueda.
Evaporación de feromonas	Reducción del nivel de feromonas presentes en los arcos.

Cuadro 4.1: Conceptos computación de enjambre

4.4.2. Línea de tiempo

En la figura 4.4 se puede observar la linea de tiempo correspondiente a varios sucesos importantes en esta area.

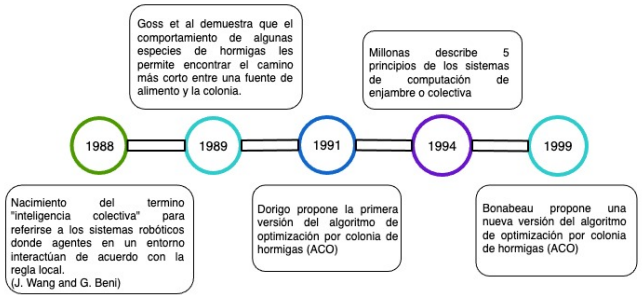


Figura 4.4: Línea de tiempo computación de enjambre

Parte III

Prácticas

Capítulo 5

Banco de retos

5.1. Introducción

En esta sección se detalla el contenido del banco de retos propuesto como recurso para apoyar el aprendizaje del curso; este banco está compuesto por preguntas de reflexión, ejercicios y problemas para cada una de las tendencias seleccionadas.

Las preguntas de reflexión buscan que el estudiante valide los conceptos teóricos obtenidos y posteriormente los use para desarrollar nuevos conceptos. Los ejercicios buscan que los estudiantes pongan en práctica los conceptos aprendidos por medio de actividades guiadas para que el estudiante pueda experimentar en la práctica lo aprendido en la teoría. Los problemas buscan que el estudiante aplique los conocimientos obtenidos bajo un contexto específico y que logre encontrar una solución adecuada para las situaciones problemáticas planteadas.

5.2. Computación neuronal

5.2.1. Preguntas

Asociadas a los videos En el video "Neurona y Aprendizaje"[8] se muestra la estructura biológica de las redes neuronales y la relación entre la actividad neuronal y las funciones de memoria y aprendizaje del ser humano. En el video "Experiencia 360° |Sinapsis, el cerebro por dentro"[70] se detalla a nivel biológico la composición y el funcionamiento de las redes neuronales. A partir de estos videos y de la teoría asociada, se plantean las siguientes preguntas:

- ¿Considera que el comportamiento de las neuronas es digno de imitar? ¿Por qué?
- ¿Tiene recuerdos asociados a sabores, olores o sonidos? ¿Por qué cree que estos se generaron?
- En el vídeo se menciona la repetición como mecanismo para reforzar las conexiones, ¿Qué otros mecanismos hay para generar conexiones a largo plazo?, ¿En qué momentos utiliza el mecanismo de la repetición?

Asociadas al simulador En el simulador "*Tensorflow playground*"[77] se presenta un modelo interactivo de red neuronal que permite al estudiante trabajar con varios *datasets* y distintas configuraciones para la red, es posible modificar la arquitectura de redes ocultas, las características de entrada y distintos hiper-parámetros. A partir de la experimentación que se realice sobre este simulador, se proponen las siguientes preguntas:

- ¿Qué efectos tiene el ritmo de aprendizaje sobre el entrenamiento? ¿Qué pasa si se selecciona un ritmo de aprendizaje demasiado grande o pequeño? ¿Qué comportamientos de la función de costo nos pueden dar indicios de que necesitamos otro valor para el ritmo de aprendizaje?
- En la teoría se menciona que añadir capas de profundidad nos permite lidiar con funciones más complejas. ¿Es siempre buena idea añadir más capas al modelo? ¿Qué efectos negativos podría tener añadir más capas de las necesarias?
- ¿Por qué es importante la correcta selección de características de entrada? ¿Considera que la selección de estas características podría llegar a ser un problema en el diseño de soluciones de este tipo?

5.2.2. Ejercicios

Asociadas al simulador A partir del simulador "*Tensorflow playground*"[77] mencionado anteriormente se plantea el siguiente ejercicio

- Tomando como punto de partida la configuración inicial del simulador, resuelva el *dataset* "*Spiral*" con el menor error posible, documente las distintas arquitecturas con las que experimenta y las razones que lo llevaron a modificar el modelo

5.2.3. Problemas

Dataset circle Es un problema de clasificación binaria muy simple, en la que se simulan los perímetros de dos círculos con centro en el origen, como los que se muestran en la figura 5.1. Se busca que a partir de las coordenadas x,y de un punto sobre el plano se pueda saber a que circulo pertenece

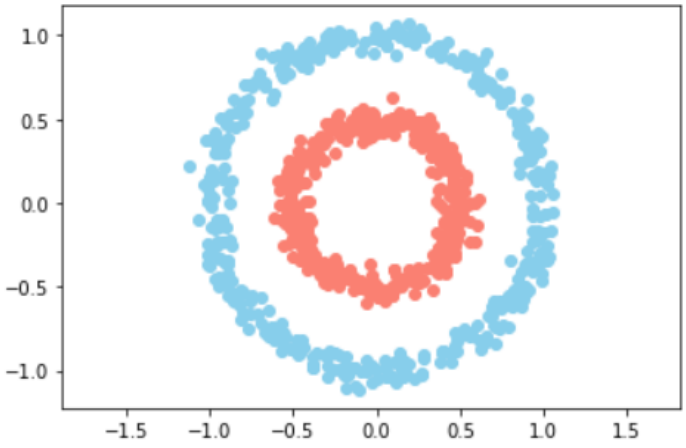


Figura 5.1: Ejemplo del *dataset circle*

***Dataset* MNIST** Es un problema de clasificación múltiple, dónde a partir un conjunto de imágenes en blanco y negro de 28x28 se busca obtener el dígito (0-9) presente en cada imagen. Es un problema histórico de reconocimiento de imágenes que ha estado presente en la evolución de las soluciones de inteligencia artificial.

Para el entrenamiento se utilizará el *dataset* MNIST, este *dataset* contiene 70.000 imágenes de las cuales 60.000 se destinan al entrenamiento de la red y 10.000 a las pruebas de la red. Todas estas imágenes están en escala de grises en un rango entre 0 y 255, dichas imágenes tienen un tamaño de 28 $\times$ 28 píxeles donde la dimensión de cada vector de la muestra es de 28 * 28 = 784. En la figura 5.2 se observan algunos ejemplos de muestras de este *dataset*.

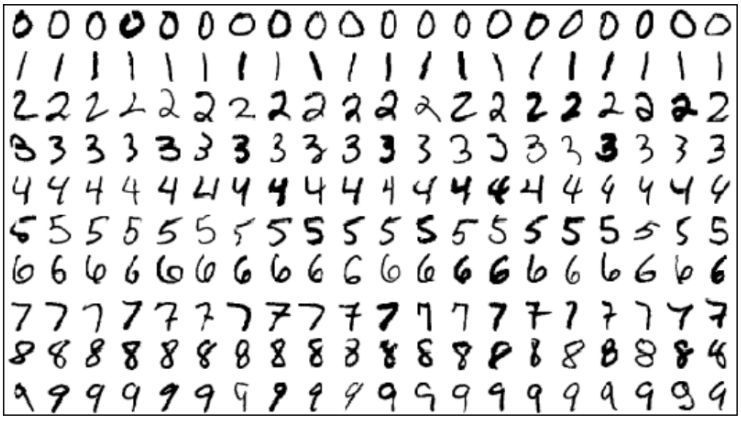


Figura 5.2: Ejemplares del *dataset* MNIST

5.3. Computación evolutiva

5.3.1. Preguntas

Asociadas a los videos En el video "Explicación biológica genética de la evolución de las especies" [53] se explican los componentes a nivel genético que dan paso a la evolución además de explicar, de forma introductoria, el proceso de selección natural. En el video "Ciencia exprés: selección natural"[29] se detalla el proceso de selección natural por medio de un ejemplo en el que se muestra la evolución de una raza de conejos, bajo ciertas condiciones de ambiente. A partir de estos videos y de la teoría asociada, se plantean las siguientes preguntas:

- ¿Por qué considera que este fenómeno es digno de imitar?
- ¿Qué elementos aleatorios tiene el proceso de evolución? ¿Qué elementos no son aleatorios? ¿En general, es el proceso de evolución un fenómeno aleatorio?
- ¿La evolución es un fenómeno que ocurre a nivel de los individuos? ¿Es un fenómeno a nivel de la población? En cualquiera de los casos ¿por qué?
- ¿De qué depende que una mutación se determine como benigna o maligna?
- ¿Teniendo en cuenta el ambiente en el que vivimos actualmente, que mutaciones cree que podrían mejorar la adaptación del ser humano?

Asociados al simulador En el simulador "Minimal Genetic Algorithm"[54] se propone un problema de búsqueda en el que a partir de un conjunto de números aleatorios de longitud N se necesita obtener el número más grande posible de N cifras (un número compuesto por N veces el dígito "9"), a partir de este problema, se puede visualizar el proceso de evolución en cada generación y experimentar con aspectos del algoritmo genético tales como : probabilidad de mutación, tamaño de la población. A partir de la experimentación que se realice sobre este simulador, se proponen las siguientes preguntas:

- ¿ Qué efecto tiene el número de la población (*turtles-number*) sobre el modelo?
- ¿Cómo influye una taza de mutación demasiado baja?¿Cómo influye una taza de mutación demasiado alta? ¿ Qué pasa si la taza de mutación es 0?
- ¿Qué condición de finalización tiene el modelo?

5.3.2. Ejercicios

Asociados al simulador A partir del simulador "Minimal Genetic Algorithm"[54] mencionado anteriormente se plantea el siguiente ejercicio

- Experimente con el problema utilizando un número de genes igual a 15, que valor de hiper-parámetros encuentra óptimos para solucionar el problema en el menor número de iteraciones posible? Anote sus experimentos y analice los aspectos que lo ayudaron a tomar decisiones.

5.3.3. Problemas

Arreglo de unos Es un problema sencillo de búsqueda en el que a partir de un conjunto aleatorio de cadenas de N binarios se necesita obtener como resultado la cadena compuesta únicamente por "1"tal y como se muestra en la figura 5.3

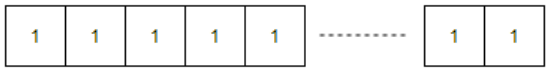


Figura 5.3: Arreglo solución para el problema *AllOnes*

El problema de las N reinas Consiste en colocar N reinas en un tablero de ajedrez de dimensiones NxN de tal forma que ninguna de las reinas pueda atacarse entre si. Es un problema clásico de combinatoria propuesto originalmente por Max Bezzel en 1848. Históricamente ha sido un problema de interés para matemáticos e informáticos debido a la gran complejidad que tiene debido a la gran cantidad de operaciones necesarias para solucionar el problema.

Algunas soluciones conocidas en la computación clásica son : (i) la búsqueda por fuerza bruta con complejidad O(N^N), y (ii) la implementación de *backtracking* que tiene una complejidad de O(2^N) . En términos de operaciones para el caso N=8, estas implementaciones cuentan con 16,777,216/ 40,320 / 256 operaciones respectivamente.

En la figura 5.4 se presentan los posibles movimientos de una reina en un tablero de 8x8 , en color rojo se muestran las posiciones que son atacadas por la reina

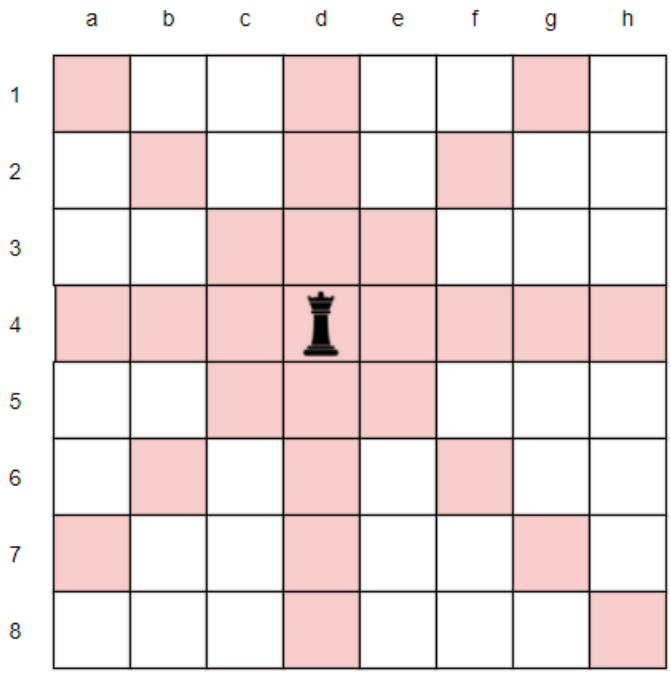


Figura 5.4: Movimientos de una reina en una tablero de 8X8

5.4. Inteligencia de enjambre

5.4.1. Preguntas

Asociadas a los videos En el vídeo "*Can ants choose the shortest path to a food source?*"[12], por medio de un experimento con hormigas reales, se muestra la capacidad de las hormigas de encontrar el camino más corto hacia una fuente de comida y se explica el uso que se le da a la feromona y la importancia de esta en la elección de caminos. A partir de este video y de la teoría asociada se plantean las siguientes preguntas:

- ¿Qué características tienen los individuos de un enjambre? ¿qué deben hacer para cumplir tareas complejas?
- ¿Para qué otras actividades cree que las hormigas utilizan las feromonas?
- ¿Qué otros insectos sociales conoce? ¿qué comportamiento grupal complejo realizan?
- ¿Por qué el comportamiento de las hormigas es digno de imitar?

Asociados al simulador En el simulador "*ACO Simulator*"[40] se utiliza el algoritmo de optimización de colonia de hormigas(ACO) para solucionar el problema de agente viajero (TSP), en este simulador se puede visualizar el recorridos de las hormigas, los niveles de feromona en cada camino y el rastreo del mejor camino en cada iteración, adicionalmente, es posible experimentar con hiper-parámetros como α ,β , la evaporación y el número de iteraciones. A partir de la experimentación que se realice sobre este simulador, se proponen las siguientes preguntas:

- ¿Qué efecto tiene el parámetro rho(evaporación) sobre el algoritmo? ¿qué pasa si es muy alto o muy bajo?
- ¿Qué hiper parámetro no está en el simulador y sería útil explorar?¿qué efectos podría agregar ese hiper parámetro?
- ¿Qué heurística de deseabilidad cree que utiliza el simulador?
- ¿Qué efecto tiene el parámetro beta (heurística) sobre el algoritmo?¿qué pasa si es muy alto o muy bajo?
- ¿Qué efecto tiene el parámetro alpha(feromona) sobre el algoritmo?¿qué pasa si es muy alto o muy bajo?

5.4.2. Ejercicios

Asociados al simulador A partir del simulador "*ACO Simulator*"[40] mencionado anteriormente se plantea el siguiente ejercicio:

- Desde la pestaña insert seleccione el ejemplo dj30.csv. A partir de esta configuración busque la configuración de hiperparámetros que le permita encontrar la menor distancia para este conjunto de puntos, mencione la lista de experimentos y los aspectos que lo llevaron a cambiar los valores de los hiperparámetros, adicionalmente mencione cual fue la configuración final, junto con la distancia obtenida y el número de iteraciones utilizadas.

5.4.3. Problemas

Encontrar punto central en una cuadricula El objetivo de este problema es encontrar el punto central de una cuadricula de dimensiones NxN , a partir de cualquier punto de la cuadrícula (El punto inicial podría ser cualquiera y se inicializa al azar), teniendo en cuenta que solo se pueden realizar movimientos verticales y horizontales. En la figura 5.5 se muestra el punto de solución para un tablero de 7x7

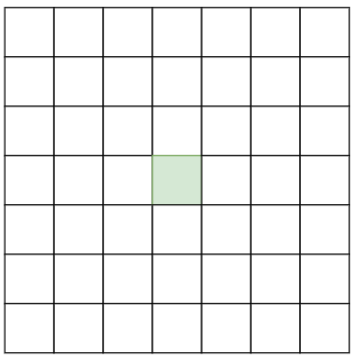


Figura 5.5: Punto solución para tablero de 7x7

El problema de la mochila Este es un problema clásico de combinatoria en el cual dada una mochila, su capacidad o peso máximo, y un conjunto de objetos con peso y valor asociados se busca la combinación de objetos que maximicen el valor de la mochila sin violar la

restricción del peso soportado por la mochila. Este problema forma parte de los 21 problemas NP-completos formulados por Richard Karp en un artículo de 1972. Este mismo ha sido intensamente estudiado por matemáticos e informáticos de todo el mundo debido a su importancia histórica al ser un problema NP-completo, siendo este un problema altamente complejo de resolver mediante mecanismos tradicionales.

Algunos enfoques tradicionales incluyen soluciones usando fuerza bruta, computación dinámica, algoritmos genéticos, etc. Donde los enfoques de computación natural destacan por sus resultados.

En la figura 5.6 se observa de forma gráfica una situación de este problema, donde se tiene una mochila que puede soportar una cantidad determinada de peso, y el objetivo es llenarla con un conjunto de objetos que aporten el valor máximo sin sobrepasar el peso limite.

Figura 5.6: Ejemplo de problema de mochila

Capítulo 6

Laboratorios

6.1. Introducción

En esta sección se presenta la metodología utilizada para la construcción de los laboratorios, y los resultados obtenidos para cada una de las tendencias: computación neuronal, computación evolutiva e inteligencia de enjambre.

6.2. Metodología

Con el objetivo de crear una experiencia de andamiaje para el estudiante, se decidió dividir el laboratorio en dos partes: (i) la implementación base de cada algoritmo y la validación con un problema simple y (ii) la solución de un problema de interés en el que se pueda utilizar el algoritmo aprendido.

Estos laboratorios serán implementados por medio de *notebooks* de python, puesto que ofrecen versatilidad al permitir la inclusión de zonas de código y zonas de texto.

A continuación se describe el contenido de las secciones mencionadas:

6.2.1. Implementación base del algoritmo

El objetivo de las zonas de implementación escogidas es que el estudiante entienda y aprenda como construir los principales componentes de cada algoritmo.

Para esta sección se utilizaron las celdas de texto para incluir la teoría necesaria para cada una de las zonas de implementación seleccionadas, cada zona de implementación corresponde a un componente del algoritmo, adicionalmente se incluyen pruebas para cada zona de implementación y así poder asegurar que el estudiante logre una implementación correcta. Adicionalmente a las zonas de teoría y de implementación también se añaden zonas de código para orquestar las zonas de implementación construidas por el estudiante.

Finalmente, la implementación se valida utilizando un problema simple, que sirva para dar un primer acercamiento al uso del algoritmo y la experimentación con los hiper-parámetros

6.2.2. Problema a resolver

El objetivo de esta sección es que el estudiante aplique los conocimientos obtenidos hasta el momento diseñando e implementando una solución al problema propuesto.

En esta sección se propone al estudiante un problema interesante, desafiante e importante para el área, en el *notebook* solo se especifica el enunciado del problema y en ocasiones el *dataset* que se utilizará para validar el problema.

6.3. Computación neuronal. Perceptrón multicapa.

El objetivo de este laboratorio es desarrollar competencias básicas para:

- Implementar los componentes básicos de una red neuronal
- Sintonizar los hiper parámetros de una red y entrenarla
- Solucionar un problema práctico utilizando un modelo de red neuronal

6.3.1. Implementación base del algoritmo

Funciones de activación

En esta sección se construye la lógica correspondiente a algunas de las funciones de activación más populares. Se debe implementar los métodos correspondientes tanto a la aplicación de cada función como a la aplicación de la derivada de cada función, las funciones que se implementarán se listan a continuación

- ReLu
- Sigmoide
- Linear
- Tangente hiperbólico

Capa completamente conectada

En esta sección los estudiantes deben construir la lógica correspondiente a una capa completamente conectada, para esto es necesario realizar la implementación de los siguientes métodos :

- Inicialización aleatoria de los parámetros (Matriz de pesos y vector de sesgo)
- Paso hacia adelante (*Forward*)
- Paso hacia atrás (*Backward*)

Funciones de costo

En esta sección se construye la lógica correspondiente a algunas de las funciones de costo más populares. Se debe implementar los métodos correspondientes tanto a la aplicación de cada función como a la aplicación de la derivada de cada función, las funciones que se implementarán se listan a continuación

- Error cuadrático medio
- Error entropía cruzada

Red neuronal

En esta sección el estudiante debe completar los métodos de propagación hacia adelante y hacia atrás realizando la invocación de los métodos implementados en la sección de capa completamente conectada.

Problema de validación

El problema de validación corresponde al *dataset circle* especificado en el banco de retos.

6.3.2. Problema a resolver

El problema a resolver corresponde al *dataset MNIST* especificado en el banco de retos. Como apoyos al estudiante se presenta la carga de datos desde el *dataset* y el pre-procesamiento necesario para tener valores entre 0 y 1.

6.4. Computación evolutiva. Algoritmos genéticos.

El objetivo de este laboratorio es desarrollar competencias básicas para:

- Implementar los componentes básicos de un algoritmo genético
- Sintonizar y experimentar con los distintos hiper parámetros de un modelo de algoritmo genético
- Solucionar un problema práctico utilizando un modelo de algoritmo genético

6.4.1. Implementación base del algoritmo

Individuo y población

En esta sección el estudiante implementar el código correspondiente a la inicialización aleatoria de un individuo basándose en un conjunto de alelos(posibles valores) y la generación de una población compuesta de dichos individuos.

Operador de selección

En esta sección se debe completar la lógica correspondiente al operador de selección de ruleta, se provee la implementación correspondiente a la construcción de la ruleta. El estudiante debe implementar las zonas correspondientes a evaluar los individuos y seleccionarlos a partir de la ruleta construida

Operador de cruce

En esta sección se debe completar el código correspondiente al operador de cruce de punto simple, el estudiante debe implementar la selección del punto de cruce, la reproducción asexual(clonación), sexual (Combinación por punto de cruce) y la elección del tipo de reproducción a partir de la probabilidad de cruce

Operador de mutación

En esta sección se debe implementar el código correspondiente al operador de mutación de reemplazo de alelo aleatorio en el que se seleccionan un conjunto de genes según la probabilidad de mutación y se cambian por otro valor en el conjunto de alelos.

Algoritmo genético

En esta zona de implementación se orquestan todos los componentes para construir el algoritmo como tal. Es importante tener en cuenta que para poder utilizar esta implementación, es necesario implementar una función de aptitud, que evalúe la bondad de cada individuo dependiendo la necesidad de cada problema

Problema de validación

El problema de validación utilizado corresponde al problema .Arreglo de unos.especificado en el banco de retos, para este problema de validación es necesario implementar una función de aptitud que cuente cuantos "1"hay en un arreglo.

6.4.2. Problema a resolver

El problema a resolver corresponde al problema de las N reinas especificado en el banco de retos

6.5. Inteligencia de enjambre. Optimización de colonia de hormigas.

El objetivo de este laboratorio es desarrollar competencias básicas para:

- Implementar los componentes básicos de un algoritmo de optimización de colonia de hormigas
- Sintonizar y experimentar con los distintos hiper parámetros de un modelo de colonia de hormigas
- Solucionar un problema práctico utilizando un modelo de colonia de hormigas

6.5.1. Implementación base del algoritmo

Ambiente

En esta sección se debe implementar el código correspondiente a inicializar el grafo o ambiente de la colonia de hormigas. La definición de este grafo depende del problema donde generalmente se representa como una matriz de distancias.

Feromona

En esta sección se debe implementar el código correspondiente a inicializar las feromonas correspondientes a cada posición en la matriz de distancia y definir la función delta la cual es responsable de depositar la cantidad de feromonas correspondientes.

Heuristica de deseabilidad

En esta sección se debe implementa el método que determina el valor de la heurística de deseabilidad de los puntos adyacentes (vecinos) a la posición actual. De esta forma se complementa la decisión sobre cual posición de la cuadricula será el siguiente en ser visitada. La función heurística propuesta para el problema está definida para que entre mayor sea la distancia de una posición al centro menor será el valor de la heurística.

ACO Grid

Esta sección está encargada de generar las soluciones para el contexto dado, utiliza el nivel de feromonas en cada arco del grafo para decidir cual arco debería recorrer en el siguiente movimiento y de esta forma construir los caminos (soluciones) .

ACO

En esta zona de implementación se orquestan todos los componentes para construir el algoritmo de colonia de hormigas

Problema de validación

El problema de validación utilizado corresponde al problema de encontrar el punto central de una cuadrícula especificado en el banco de retos.

6.5.2. Problema a resolver

El problema a resolver corresponde al problema de la mochila especificado en el banco de retos.

Conclusiones y trabajo futuro

6.6. Conclusión

La creación del diseño de aprendizaje de un curso introductorio de computación natural representa un aporte valioso a la formación de los estudiantes de ingeniería de sistema puesto que, gracias a este, el estudiante obtiene una perspectiva interdisciplinar para solucionar problemas, algo que es muy valorado en la actualidad. En este momento el diseño se desplegó en una aula virtual donde se define la estructura del curso y se guía el proceso de aprendizaje en las distintas etapas diseñadas en el curso.

Durante el diseño se observó que es necesaria la correcta integración de actividades, roles y recursos para crear experiencias de aprendizaje exitosas. Adicionalmente, se evidenció la importancia de contar con diferentes tipos de recursos que generen el andamiaje necesario en el proceso de aprendizaje. Los recursos propuestos fueron: texto guía, audiovisuales (videos y simuladores), *framework* y laboratorios. Para obtener estos recursos propuestos, basados en la revisión de estado de arte, se clasificaron los recurso en los que se podrían obtener mediante curaduría (audiovisuales) y los que deberían se desarrollados (los otros recursos).

Durante el proceso de evaluación nos permite afirmar que (i) el diseño de aprendizaje es el adecuado para el logro de los objetivos; (ii) los recursos del curso permiten las experiencias de aprendizaje propuestas en el diseño; (iii) es necesario hacer un ciclo de mejoramiento sobre los recursos, teniendo en cuenta los comentarios de los revisores.

En cuanto a los recursos obtenidos por la curaduría se concluye lo siguiente: (i) el conjunto de vídeos seleccionados facilitan el entendimiento de los conceptos biológicos de cada tendencia al mismo tiempo que generan mucho interés en los estudiantes hacia el área; (ii) los simuladores de computación neuronal e inteligencia de enjambre permiten al estudiante explorar fácilmente los componentes principales de cada algoritmo, en el caso de computación evolutiva no fue posible encontrar un simulador que cumpliera las expectativas planteadas. En general, se resalta que este tipo de recursos facilitan el aprendizaje mediante la explicación y experimentación de distintos conceptos de forma gráfica.

En cuanto a los recursos desarrollados podemos concluir lo siguiente: (i) la primera versión del texto guía de computación bioinspirada escrito para estudiantes de pregrado, en el que se destaca que es un libro en español, contiene los conceptos introductorios a nivel biológico e informático necesarios para que los estudiantes logren resolver problemas en el área; (ii) el banco de retos – compuesto por preguntas de reflexión, ejercicios y problemas para cada una de las tendencias seleccionadas – son útiles para validar el conocimiento e incentivar la obtención de nuevo conocimiento; (iii) el conjunto de laboratorios para cada una de las tendencias – en el que se incluyen zonas de explicación, zonas ya implementadas y zonas por implementar – permite una experiencia de andamiaje, donde inicialmente se ofrecen ayudas al estudiante y se van retirando progresivamente para que finalmente el pueda solucionar retos interesantes de forma independiente; (iv) los *frameworks* desarrollados facilitan la implementación y construcción de soluciones usando modelos de computación bioinspirada por su diseño simple y flexible que hace que su uso sea intuitivo.

6.7. Trabajos Futuros

A continuación se mencionan los cinco trabajos futuros propuestos para este proyecto: (i) iterar nuevamente sobre los productos creados, a partir de los comentarios recibidos por los revisores, para fortalecer y mejorar la calidad de los recursos de aprendizaje; (iii) enriquecer

el banco de retos en todos sus niveles, para poder abarcar más conceptos y prácticas que puedan fortalecer los conocimientos del estudiante; (iii) desarrollar un simulador propio de algoritmos genéticos con las características deseables mencionadas en la curaduría, puesto que no fue posible encontrar un simulador que cumpliera con todas las expectativas; (iv) evaluar nuevamente el curso de forma completa con más estudiantes para obtener más comentarios y de esta forma continuar con el proceso de mejora continua; (v) ofrecer el curso como una alternativa a la materia Ciencias Naturales y Tecnología (CNYT) en la Escuela Colombiana de Ingeniería Julio Garavito.

Agradecimientos

Queremos agradecer a todos los profesores y demás personas que nos estuvieron acompañando en nuestro pregrado de ingeniería de sistemas. Empezando agradeciendo especialmente al Dr. Oswaldo Castillo Navetty por su acompañamiento durante todo el pregrado, brindándonos sus valiosos consejos los cuales siempre fueron una guía muy valiosa; además, agradecer su amistad y apoyo incondicional.

También queremos agradecer a nuestra directora de proyecto María Irma Díaz Rozo, quien durante el pregrado siempre nos brindó sus valiosos conocimientos, siendo una inspiración y apoyando de manera muy importante a nuestra formación como profesionales y personas, especialmente durante el énfasis en Inteligencia Artificial. Sin su apoyo y acompañamiento no seríamos los profesionales que somos hoy en día.

Agradecer a nuestras familias ya que sin el apoyo brindado durante todos nuestros estudios este momento no sería posible, gracias por todo el amor y el esfuerzo realizado. También queremos agradecer a todos nuestros compañeros en la universidad ya que todos los consejos, ayudas brindadas y su valiosa amistad significaron mucho durante estos años de estudio.

Por esto dedicamos este trabajo a todas estas personas que aportaron a que este trabajo fuera posible, gracias por todo.

Apéndices

Apéndice A

Presentación a decanatura

ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO

Introducción a la Computación Bioinspirada: Diseño de asignatura y creación de material de aprendizaje

Juan Camilo Angel Hernandez, Juan Camilo Rojas Ortiz

Maria Irma Diaz Rozo

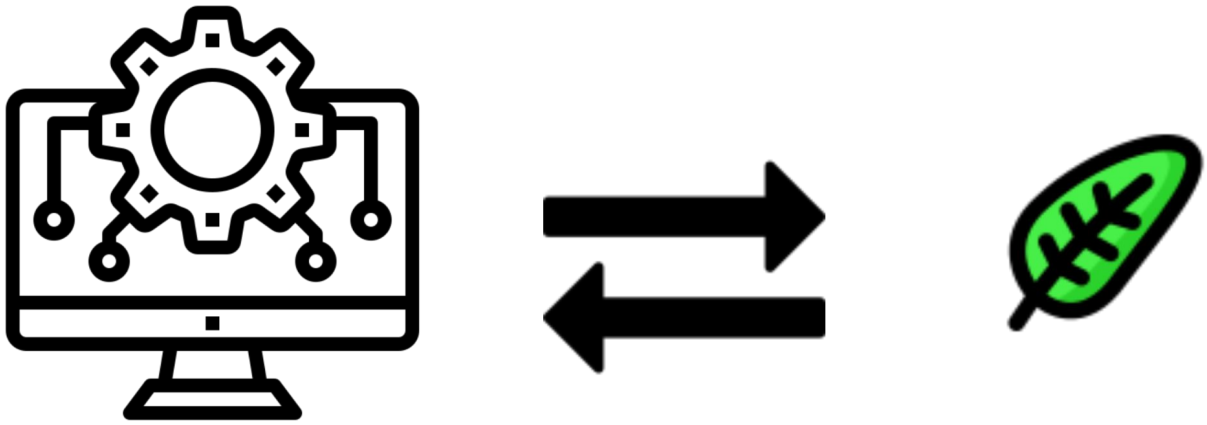
Agenda

- Introducción
- Problema
- Idea de solución
- Trabajos relacionados
- Resultados
- Conclusiones y reflexiones
- Trabajo futuro

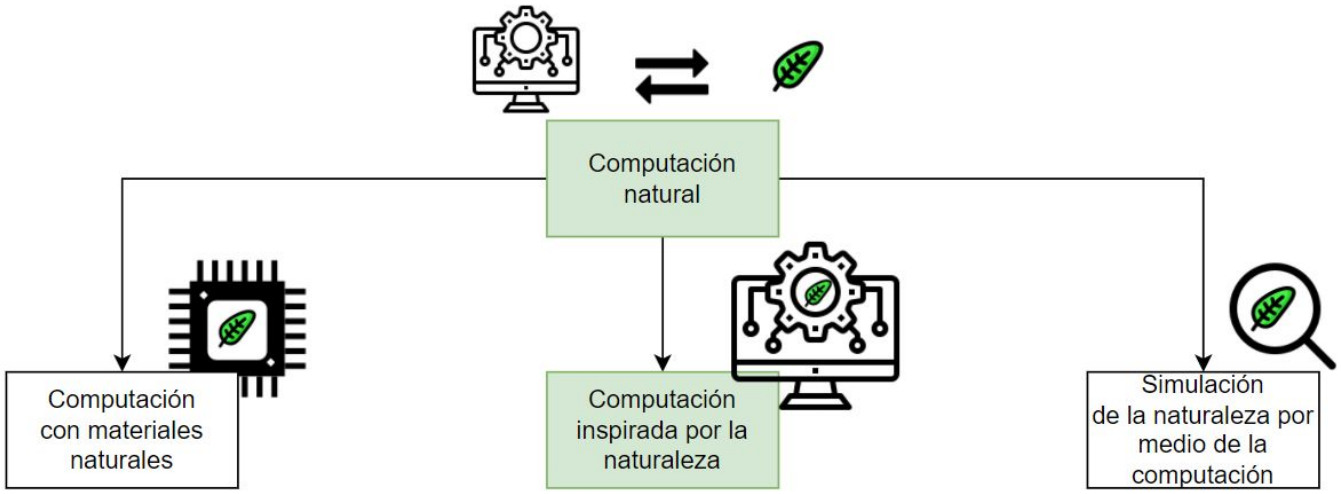
Introducción

- ¿Qué es computación natural?
- ¿Por qué nos interesan las soluciones bioinspiradas?
- ¿Cuáles son los logros alcanzados por los algoritmos bioinspirados?

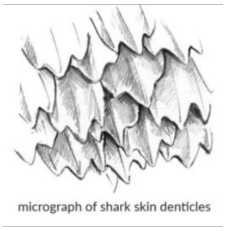
¿Qué es computación natural?

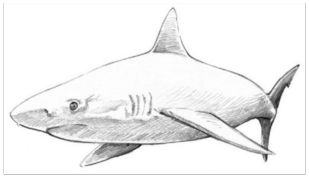


¿Qué es computación natural?



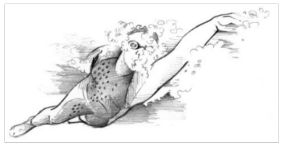
¿Por qué nos interesan las soluciones bioinspiradas?


micrograph of shark skin denticles




Sharklet™ antibacterial surface


Speedo fastskin swimsuit



¿Cuáles son los logros alcanzados por los algoritmos bioinspirados?

Presentan una inteligencia artificial que crea música y compone como Los Beatles, Mozart o Lady Gaga

MuseNet es una red neuronal que genera por su cuenta canciones en MIDI, con diferentes estilos y hasta diez instrumentos.

Fecha de publicación: **29 de abril 2019, 13:15hs**

¿La música puede ser creada por máquinas?

Computación

La IA que evoluciona y muta podría desbancar al aprendizaje profundo

La computación evolutiva, inspirada en la propia evolución, ha sido ignorada frente a la popularidad de las redes neuronales. Pero esta investigación demuestra que podría ser igual o incluso más útil en algunas tareas, y ya lo ha demostrado en algunos videojuegos. Además, es más fácil de entrenar

por Emerging Technology From The Arxiv | traducido por Mariana Díaz

INVESTIGACIÓN

IA: Un algoritmo en búsqueda de fármacos es inspirado por hormigas faraón

El objetivo del equipo de investigación es aplicar la IA en la búsqueda de fármacos y la optimización de la gestión logística, entre otros sectores

Por **Gaceta Médica** - 23 diciembre 2020

La inteligencia artificial cada vez abarca más opciones en la búsqueda de optimización de procesos. Un equipo de investigadores del Consejo Superior de Investigaciones Científicas (CSIC) se ha inspirado en el **comportamiento de las hormigas faraón** para mejorar un algoritmo de inteligencia artificial. El objetivo del equipo de investigación es aplicar la IA en la **búsqueda de fármacos y la optimización de la gestión logística**, entre otros sectores. Es posible a través de las hormigas faraón porque esta especie posee un comportamiento que les permite aprender de ejemplos negativos.

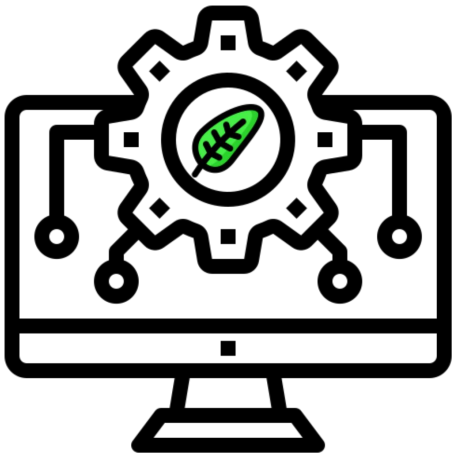
1. Presentan una inteligencia artificial que crea música y compone como Los Beatles, Mozart o Lady Gaga
2. La IA que evoluciona y muta podría desbancar al aprendizaje profundo
3. IA: Un algoritmo en búsqueda de fármacos es inspirado por hormigas faraón

Problema

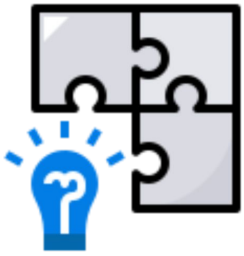
- ¿Cuál es el problema?
- ¿Por qué es importante resolverlo?

¿Cuál es el problema?

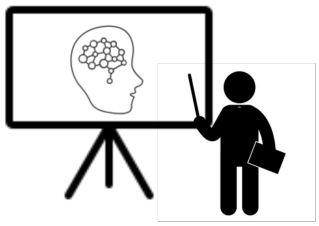
- Hay muy pocos profesionales formados en el área
- No existen cursos con las siguientes características:
  - Para estudiantes de pregrado
  - Profundidad y anchura temática adecuada
  - Enfocado a la solución de problemas
  - Recursos en español



¿Por qué es importante resolverlo?



Conocimiento útil para solucionar problemas



Introducir de manera oportuna al estudiante en el área

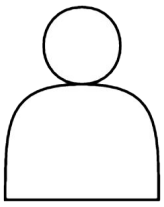
Atraer más estudiantes al área

Idea de solución

- ¿Cuál fue la idea de solución?
- ¿Cuál fue la metodología de solución?

¿Cuál fue la idea de solución?

Desarrollar un curso con las siguientes características:



Perfil: Estudiantes de pregrado

Contenido en español

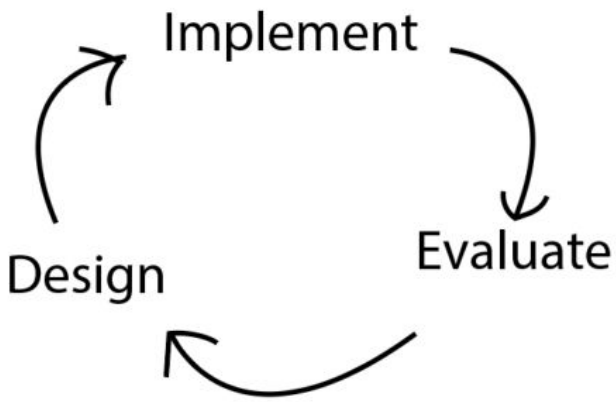
Objetivo: Introducir a los estudiantes en varias tendencias de la computación natural

Uso de apoyos visuales para facilitar el aprendizaje

Uso de laboratorios y ejercicios prácticos

¿Cuál fue la metodología de solución?

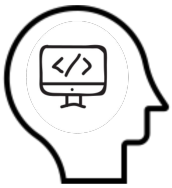
- Revisión y análisis del estado del arte
- Creación de diseño de aprendizaje
- Creación y/o selección de recursos de aprendizaje
- Despliegue de recursos en ambiente virtual (Moodle)
- Evaluación y mejora continua



Trabajos relacionados

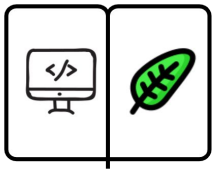
- ¿Qué aspectos generales se encontraron?
- ¿Qué se ha hecho para solucionar el problema?

¿Qué aspectos generales se encontraron?



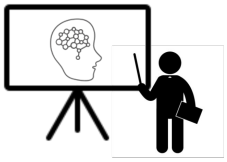
Principal requisito: saber programar

Cursos de agrado e interés



Conocimiento informático y biológico

Ejercicios de programación



Búsqueda, análisis y presentación de artículos

Uso de simuladores (pocos casos)

¿Qué se ha hecho para solucionar el problema?

	Pregrado	Español	Solución de problemas	Profundidad y anchura
• A 'Hands on' Strategy for Teaching Genetic Algorithms to Undergraduates.	✓	✗	✓	✗
• Undergraduate Course: Natural Computing (INFR11161). University of Edinburgh.	✓	✗	?	✓
• Cursos: Inteligencia Artificial.Fernando Sancho Caparrini.	?	✓	✓	✗
• A Virtual Laboratory on natural computing: A Learning experiment.	✓	✗	✗	✓

1.Anne Therese Venables and Grace Tan. A 'hands on' strategy for teaching genetic algorithms to undergraduates. Journal of Information Technology Education: Research, 6, 2007
2. THE UNIVERSITY of EDINBURGH. Undergraduate course: Natural computing (infr11161). http://www.drps.ed.ac.uk/18-19/dpt/cxinfr11161.htm. Accessed on 2021-08-19
3.Fernando Sancho Caparrini. Cursos: Inteligencia artificial. http://www.cs.us.es/~fsancho/?p=inteligencia-artificial. Accessed on 2021-08-19.
4.L. Castro. Fundamentals of natural computing - basic concepts, algorithms, and applications. In Chapman and Hall / CRC computer and information science series, 2006

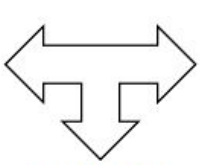
Resultados

- Creación de diseño de aprendizaje
- Creación y selección de los recursos de aprendizaje
- Despliegue de recursos en ambiente virtual (Moodle)
- Evaluación y mejora continua

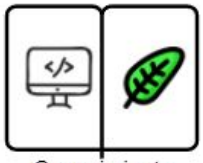
Creación de diseño de aprendizaje

- ¿Qué lineamientos dirigen el curso?
- ¿Qué temas se incluirán en el curso?
- ¿Cuáles son las experiencias de aprendizaje?

¿Qué lineamientos dirigen el curso?



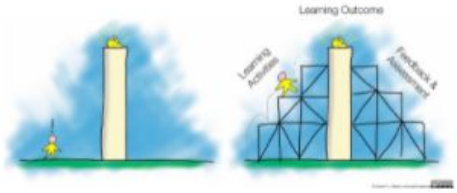
Profundidad
y anchura



Conocimiento
biológico, informático
y metodológico

Solución
de problemas

Trabajo en equipo

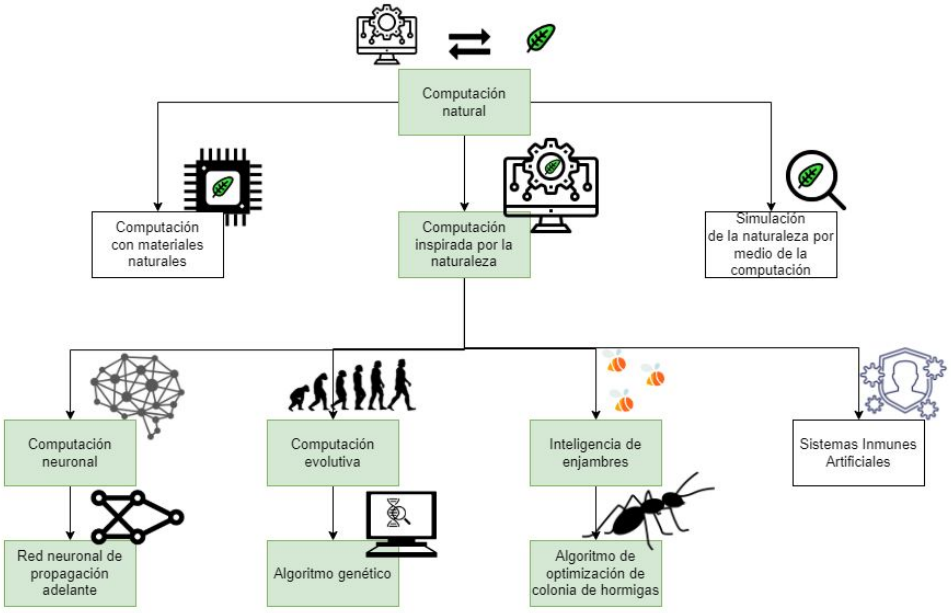


Recursos que generen andamiaje

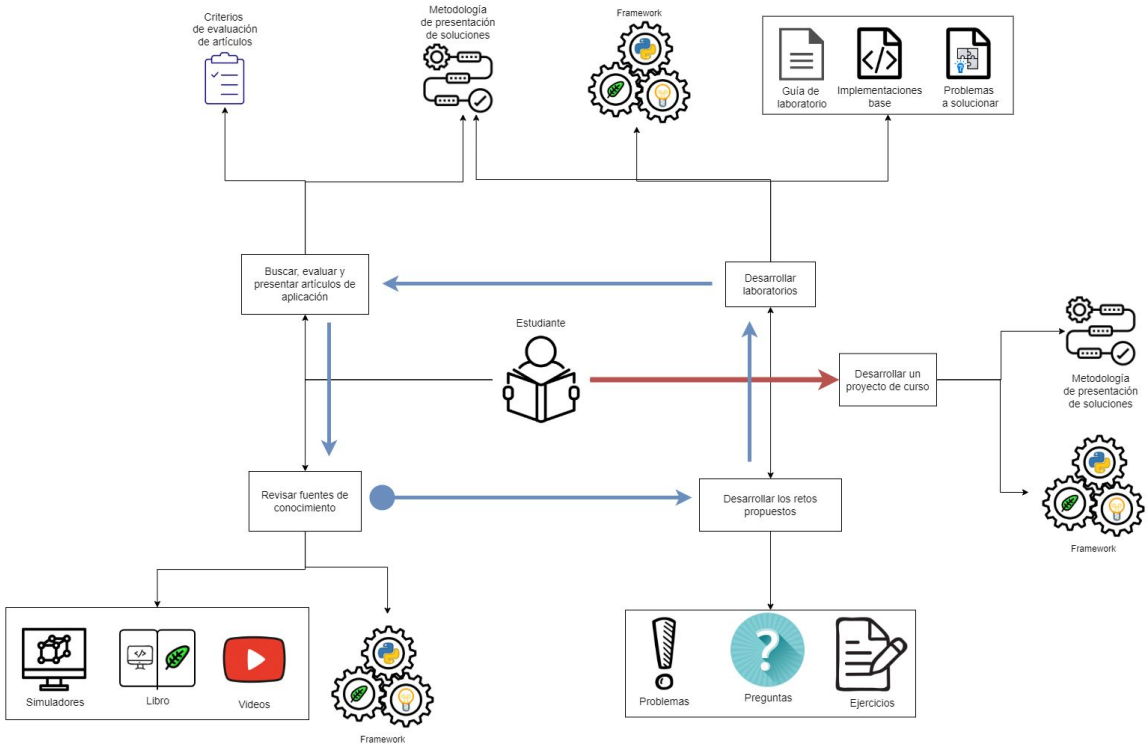
Retrospectiva para
afianzar conocimientos

Pensamiento
Crítico

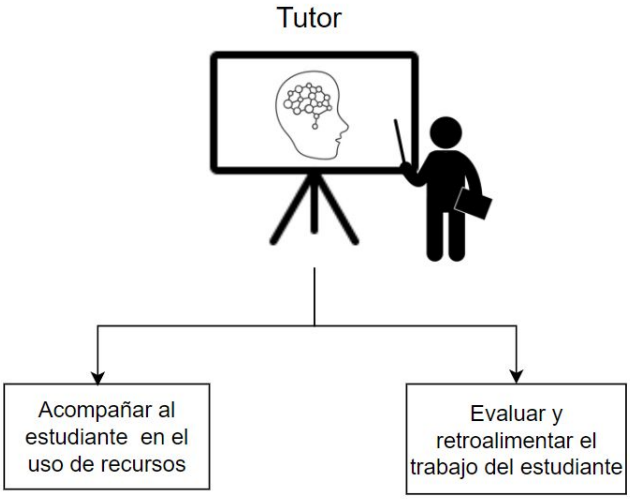
¿Qué temas se incluirán en el curso?



¿Cuáles son las experiencias de aprendizaje?



¿Cuáles son las experiencias de aprendizaje?

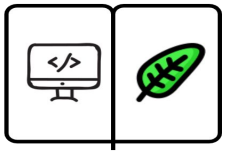


Creación y selección de los recursos de aprendizaje

- ¿Cuáles recursos se requerían?
- ¿Qué características se buscaron en los recursos?
- ¿Cómo se obtuvieron los recursos?
- ¿Qué se obtuvo de la creación/búsqueda de recursos?

¿Cuáles recursos se requerían?

Teoría



Libro

Videos

Banco de retos

Práctica

Laboratorios

Simuladores

Framework

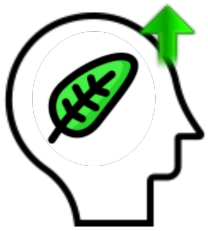
Metodología

Metodología
de presentación
de soluciones

Criterios
de evaluación
de artículos

¿Qué características se buscaron en los recursos teóricos?

Genere interés en los estudiantes



Facilite el entendimiento de los conceptos biológicos

Facilite el entendimiento de los conceptos informáticos

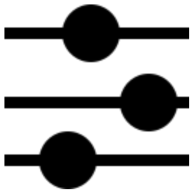
Muestre de forma fiel cada fenómeno

Presente los conceptos de forma corta pero concisa y clara

¿Qué características se buscaron en los recursos prácticos?

Genere interés en los estudiantes

Facilite el entendimiento de los conceptos informáticos



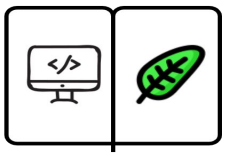
Permita explorar el uso de los hiper parámetros

Permita solucionar varios problemas

Fácil e intuitivo de usar

¿Cómo se obtuvieron los recursos?

Creación



Libro

Banco de retos

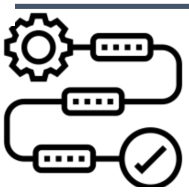
Curaduría

Videos

Simuladores

Creación

Laboratorios



Metodología
de presentación
de soluciones

Framework

Criterios
de evaluación
de artículos

¿Qué se obtuvo de la creación del libro?

Computación evolutiva | **3**

3.1 Introducción

A lo largo de la historia de nuestro planeta se ha observado la enorme capacidad que tienen los seres vivos para adaptarte al cambio presente en el entorno donde habitan. Se han observado organismos que han desarrollado características especiales según las condiciones del entorno, donde pueden haber condiciones extremas, escasez de recursos, etc, de esta manera se han adaptado y logrado sobrevivir. Esta capacidad de evolución y adaptación ha llamado la atención de investigadores de diferentes áreas donde se busca imitar estos comportamientos biológicos a la resolución de problemas.

El constante cambio en el entorno y ambiente es algo que resulta en la muerte de los individuos menos adaptados a su entorno, así como en la supervivencia de los mejor adaptados, este comportamiento conocido como selección natural impulsa la diversidad biológica, comportamiento que se repite a lo largo de las generaciones. Dicha diversidad se refuerza aun mas con la aparición de mutaciones en los organismos que pueden tener resultados positivos en la adaptabilidad del individuo, esta diversidad biológica involucra cambios en los organismos que son heredados hacia las futuras generaciones mediante la reproducción, de esta manera se está garantizando una mejor adaptabilidad y supervivencia de la especie ya que las futuras generaciones heredarán las características de organismos aptos, logrando de esta manera la supervivencia de la especie en un entorno coexistiendo con otras especies. Este proceso de adaptabilidad y cambio de los individuos a lo largo de las generaciones se conoce como evolución.

Este comportamiento observado donde los individuos mejor adaptados tienen mas posibilidades de sobrevivir y llegar a evolucionar ha inspirado un tipo de algoritmos conocidos como algoritmos evolutivos. Estos algoritmos son capaces de dar solución a problemas representando posibles soluciones como individuos e imitando el proceso de evolución esperando que luego de un número de generaciones aparezcan individuos que representen soluciones óptimas.

Algoritmos

Finalmente a continuación se pueden observar los algoritmos correspondiente al modelo de una red neuronal multicapa.

Algorithm 1: Forward propagation
Data: Input X
Result: activation value
for $l; l \leftarrow 1$ **to** $(numLayers)$ **do**
  $z^l \leftarrow W^{[l]} * a^{[l-1]} + b^{[l]}$;
  $a^l \leftarrow g^l(z^l)$

Algorithm 2: Backward propagation
Data: error
Result: gradients
$\delta^{[numLayers]} \leftarrow (y-y)g^{[l]\prime}(z^{[l]})$;
for $l; l \leftarrow (numLayers-1)$ *downto* 1 **do**
  $\delta^{[l]} \leftarrow (\delta^{[l+1]} * w^{[l]})g^{[l]\prime}(z^{[l]})$;

Algorithm 3: train MLP
Data: dataset
Result: trained parameters
Inicializar los pesos y el bias;
for $i; i \leftarrow 1$ **to** $(numEpochs)$ **do**
  do forward propagation;
  get loss;
  $\delta^{[l]} \leftarrow backwardpropagation$
  for $l; l \leftarrow 1$ **to** $numLayers$ **do**
    $W^{[l]} = W^{[l]} - \eta * a^{[l]} * \delta^{[l]}$

2.2 Fundamentos biológicos

La neurona

La neurona es una célula nerviosa presente en el cerebro, pero es diferente a la mayoría de las células que conocemos: no tienen un mecanismo de reproducción y tienen la capacidad de enviar y procesar información done la información aquí está representada mediante impulsos eléctricos o reacciones químicas. Las neuronas al no reproducirse son finitas; pero esta limitación es cubierta gracias a la plasticidad que se logra mediante la creación o eliminación de conexiones que antes no existían.

Una neurona tiene tres componentes principales, dendritas(zonas de recepción), un soma (zona de procesamiento), y un axón (zona de transmisión). Una neurona tiene varias dendritas y cada una de ellas recibe la señal enviada por la neurona a la cual está conectada. El soma, el cuerpo de la neurona, es la parte encargada de realizar la agregación de las señales de entrada recibidas por las dendritas y cuando esta alcanza cierto umbral dispara una señal de salida. El axón es la parte encargada de enviar los impulsos de salida hacia una dendrita perteneciente a otra neurona, formando de esta manera una sinapsis o conexión. En la figura 2.1 podemos observar la representación gráfica de una neurona y sus componentes.

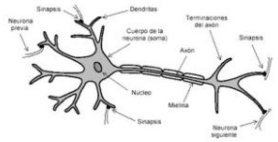


Figure 2.1: Estructura de una neurona [

¿Qué se obtuvo de la creación de los laboratorios?

Sigmoide

A continuación se presenta la formula de sigmoide, para su implementación es necesario utilizar el metodo $np.exp()$

$$sigm(z) = \frac{1}{1 + e^{-z}}$$

La formula de la derivada de la sigmoide se define a continuación

$$sigm'(z) = sigm(z) * (1 - sigm(z))$$

```
class Sigmoid:
  def apply(self, z):
    # aprox 1 linea (aplicar sigmoide a z)
    # return

  def df(self, z):
    # aprox 2 lineas (calcular el valor de derivada dado z)
    # y =
    # return
```

¿Qué se obtuvo de la creación de los laboratorios?

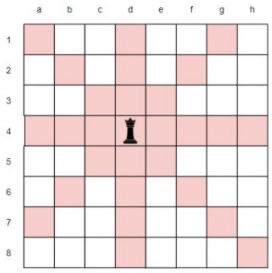
Descripción del problema

El problema de las N reinas

El objetivo de este laboratorio es resolver el problema de las N reinas, es un problema clásico de combinatoria, consiste en colocar N reinas en un tablero de ajedrez de dimensiones NxN de tal forma que ninguna de las reinas pueda atacarse entre si. Este problema fue propuesto originalmente por Max Bezzel en 1848. Históricamente ha sido un problema de interés para matemáticos e informáticos debido a la gran complejidad que tiene debido a la gran cantidad de operaciones necesarias para solucionar el problema.

Algunas soluciones conocidas en la computación clásica son : la búsqueda por fuerza bruta con complejidad $O(N^N)$, si en esta implementación se evita colocar dos reinas en la misma fila la complejidad se convierte en $O(N!)$, finalmente se tiene la implementación de backtracking que tiene una complejidad de $O(2^N)$. En términos de operaciones para el caso N=8, estas implementaciones cuentan con 16,777,216/ 40,320 / 256 operaciones respectivamente.

A continuación se presentan los posibles movimientos de una reina en un tablero de $8x8$, en color rojo se muestran las posiciones que son atacadas por la reina



Descripción del problema

El objetivo de este laboratorio es solucionar un problema de clasificación múltiple, especificamente, se busca clasificar correctamente el dígito correspondiente a una imagen en blanco y negro de 28x28 pixeles.

MNIST

A continuación se cargan los datos del dataset MNIST. este dataset contiene 70000 imágenes de dígitos escritos a mano, de las cuales 60.000 se destinan al entrenamiento de la red y 10.000 a las pruebas de la red. Todas estas imágenes están en escala de grises en un rango entre 0 y 255, dichas imágenes tienen un tamaño de 28 × 28 pixeles, por lo que la dimensión de cada vector de la muestra es de 28 * 28 = 784

```
[ ]
```

```
[ ] i = 12

    plt.imshow(x_train[i,:].reshape(28,28), cmap = matplotlib.cm.binary)
    plt.axis("off")
    plt.show()
    print("Imagen de un número", np.argmax(y_train[i,:]))
```

Imagen de un número 3

El Problema de la mochila (Knapsack problem)

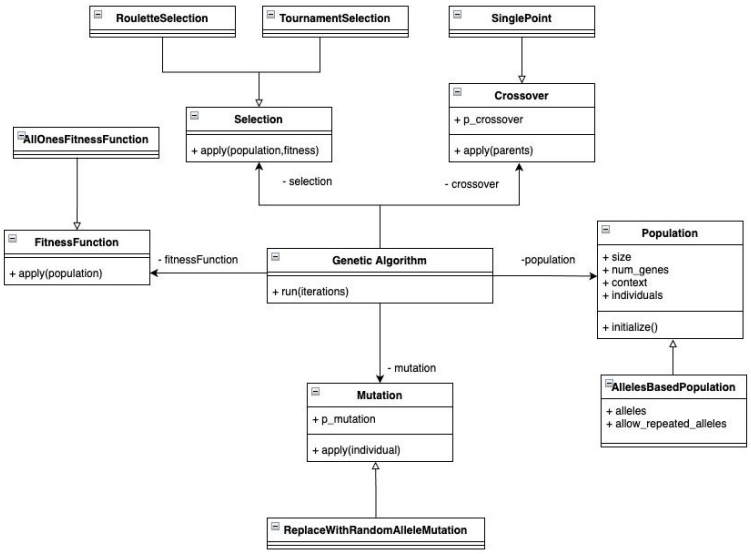
El objetivo de este laboratorio es resolver el problema de la mochila, este es un problema clásico de combinatoria en el cual dada una mochila, su capacidad o peso maximo, y un conjunto de objetos con peso y valor asociados se busca la combinacion de objetos que maximen el valor de la mochila sin violar la restricción del peso soportado por la mochila. Este problema forma parte de los 21 problemas NP-completos formulados por Richard Karp en un articulo de 1972. Este mismo ha sido intensamente estudiado por matematicos e informaticos de todo el mundo debido a su importancia historica al ser un problema NP-completo, siendo este un problema altamente complejo de resolver mediante mecanismos tradicionales.

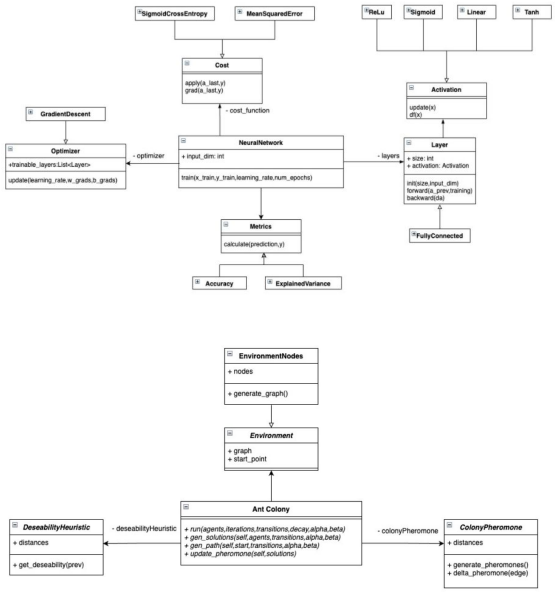
Algunos enfoques tradicionales incluyen soluciones usando fuerza bruta, computación dinamica, algoritmos geneticos, etc. Donde los enfoques de computación natural destacan por sus resultados.

En la siguiente imagen se observa de forma grafica una situación de este problema, donde se tiene una mochila que puede soportar una cantidad determinada de peso, y el objetivo es llenarla con un conjunto de objetos que aporten el valor máximo sin sobrepasar el peso limite.

Dado esto este problema se puede representar mediante una lista de tuplas indicando el peso y el valor de cada articulo. $[(v_1, p_1), (v_2, p_2), \ldots, (v_n, p_n)]$ siendo v y p el valores y pesos correspondientes. De esta forma una solución a este problema está formada por un conjunto de articulos.

¿Qué se obtuvo de la creación del framework?





¿Qué se obtuvo de la creación del framework?

Formulación de la solución

¿Cual es el problema?

El problema a resolver es un problema conocido como el problema del agente viajero. El objetivo aqui es dada una lista de ciudades y las distancias entre cada par de ellas, encontrar la ruta más corta posible que visita cada ciudad exactamente una vez y al finalizar regresa a la ciudad origen.

Se busca resolver dicho problema usando una implementacion de un algoritmo genetico.

¿Por qué es importante resolverlo?

Resolver este problema es importante ya que este se considera como un problema NP-Hard, siendo este muy importante en ciencias de la computación porque resolverlo mediante algoritmos tradicionales es inviable al tener una complejidad alta.

¿Como se propone solucionarlo?

La propuesta de solución consiste en implementar un agoritmo genetico junto con sus operadores de manera que este sea capaz de encontrar soluciones optimas al problema del viajero, el diseño de la propuesta consiste en:

1. Operador de cruce de punto simple
2. Operador de selección por torneo
3. Mecanismo de reemplazo directo

> El diseño de las funciones de aptitud y mutación son especiales de acuerdo al problema las cuales presentan ciertas heuristicas, respecto a la función de aptitud esta utiliza una metrica de distancia la cual es la distancia de manhattan.

Finalmente los parametros de probabilades de seleccion y cruce se elegirán mediante experimentación.

¿Cómo implementaron la solución?

Primeramente se definió la representación de los individuos, un individuo está compuesto por n valores siendo n la cantidad de ciudades en el problema, dichos valores están en un arreglo y representan la secuencia de ciudades que el viajero recorre, representando de esta forma una solución.

La función de aptitud funciona realizando el calculo de $\frac{1}{distancia}$ para cada individuo, donde la $distancia$ representa la distancia total del camino que recorre el viajero, distancia la cual se quiere minimizar. En el caso de que el individuo representado presente valores repetidos este es penalizado ya que no es una solucion valida.

¿Que resultados se obtuvieron?

Se obtuvieron los siguientes resultados:

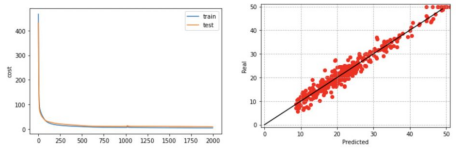
Costos de los conjuntos de entrenamiento y prueba

> 4.15 y 9.33

Varianza explicada del conjunto de prueba

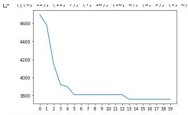
> 0.87

A continuación podemos observar la grafica de costo y la grafica correspondiente a los precios reales vs predichos por el modelo.



¿Cuáles fueron las lecciones aprendidas?

El prepocesamiento aplicado a los datos impacta de forma significativa al rendimiento del modelo al igual que la función de activación ya que al usar relu en las capas ocultas el modelo se comporta indebidamente.



shortest_path: [[32, 6], (6, 4), (4, 5), (5, 11), (11, 3), (3, 2), (2, 13), (13, 1), (1, 8), (8, 7), (7, 10), (10, 0), (0, 9), (9, 12)], 3738.0

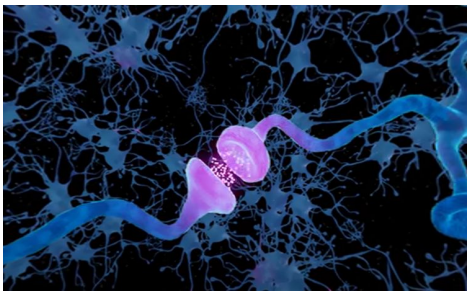
¿Que resultados se obtuvieron?

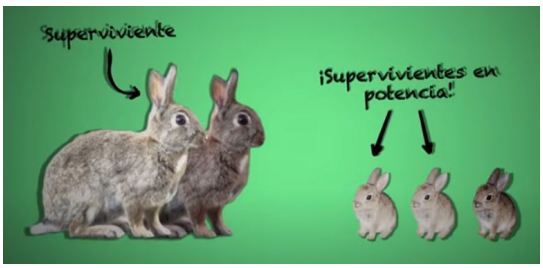
Se realizaron una serie de pruebas utilizando un conjunto de datos de varios problemas del viajero, en los primeros se cuentan con varios casos donde el numero de ciudades es 6, y en estos problemos los resultados fueron positivos ya que en la mayoria de ocasiones el algoritmo es capaz de converger en soluciones optimas resultando en la mejor solucion posible. Luego se realizó una prueba con un caso donde el numero de ciudades es 14, en este la mejor solución del modelo fue una donde la distancia total es de 3738 siendo esto un buen resultado ya que la mejor solución global es de 3323, lo anterior muestra una solidez y eficacia del modelo de algoritmo de colonia de hormigas al ser capaz de proveer soluciones optimas frente a un problema de gran complejidad como lo es este.

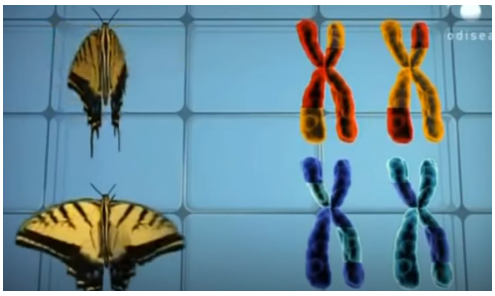
¿Cuáles fueron las lecciones aprendidas?

En el proceso de experimentación se observo el distinto impacto de los parametros donde un valor de alpha o beta tanto alto como bajo impacta en el rendimiento del algoritmo y la calidad de las soluciones.

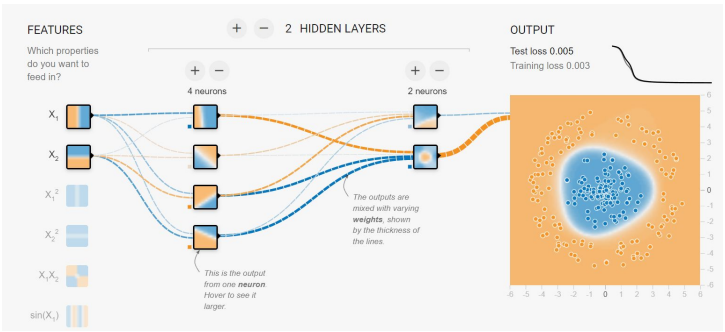
¿Qué se obtuvo de la búsqueda de vídeos?

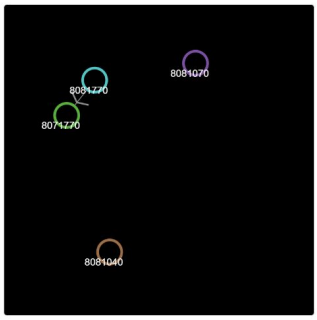


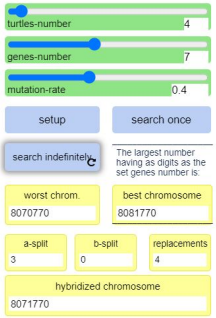


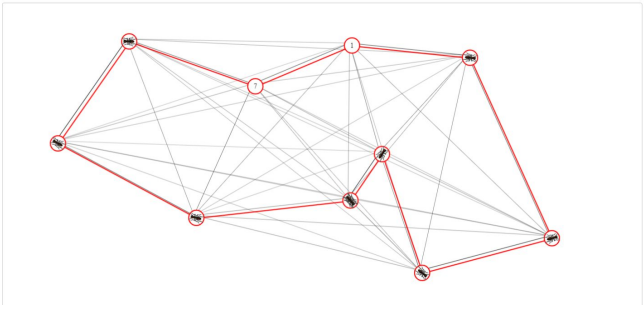


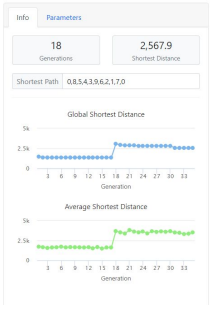
¿Qué se obtuvo de la búsqueda de simuladores?











Despliegue de recursos en ambiente virtual (Moodle)

- ¿Qué estructura se utilizó para desplegar los recursos?

¿Qué estructura se utilizó para desplegar los recursos?

Teoría biológica

REDES NEURONALES BIOLÓGICAS

LECTURA

Redes neuronales biológicas

VIDEOS

Neurona y Aprendizaje

Experiencia 360° | Sinapsis, el cerebro por dentro

DISCUSIÓN

Reflexiones sobre las redes neuronales biológicas

Teoría informática

REDES NEURONALES ARTIFICIALES

LECTURA

Redes neuronales artificiales

SIMULADOR

Tensorflow playground

DISCUSIÓN

Reflexiones sobre las redes neuronales artificiales

Ejercicio: Experimentando con el simulador

Práctica y evaluación

FRAMEWORK

Documento con diseño y api

Problema de ejemplo: Boston Houses Price

LABORATORIO

Cuía de laboratorio

Implementación algoritmo base

Problema propuesto: MNIST

Laboratorio redes neuronales

Evaluación del modulo de redes neuronales

Evaluación y mejora continua

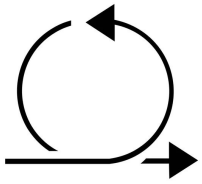
- ¿Qué estructuras de mejora continua se implementaron?
- ¿Cómo evaluaron los estudiantes el curso?

¿Qué estrategias de mejora continua se implementaron?

Revisión del equipo

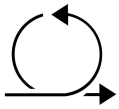
Revisión de estudiantes (2)

Encuesta de evaluación de recursos



Iterar a partir del feedback

¿Cómo evaluaron los estudiantes el curso?

		
Estructura 	Interesante, facilita el aprendizaje, hace el curso fluido	Evaluar la inclusión de pruebas calificables para validar conocimiento
Libro	Interesante, buena claridad en los capítulos de inteligencia de enjambre y computación evolutiva	Mejorar redacción de algunas secciones y equilibrar las fortalezas, complementar secciones de metodología,incluir más imágenes
Videos	Útiles para reforzar conceptos, atractivos e interesantes	-
Simuladores	Interesantes para entender los algoritmos	El simulador de algoritmos genéticos es difícil de usar, mejorar los ejercicios que acompañan la experimentación.
Framework	Útil para aprender a solucionar problemas, extensible,interesante	Complementar documentación , mejorar análisis de resultados en problemas resueltos
Laboratorios	Experiencia guiada, información necesaria al alcance, interesantes	Diseñar pruebas que faciliten detectar errores, incluir recursos interactivos,mejorar redacción de algunos fragmentos

Conclusiones y Reflexiones

- ¿Qué podemos aprender de los resultados?
- ¿Cuáles son las contribuciones principales?
- ¿Cuáles son los trabajos futuros?

¿Qué podemos aprender de los resultados?

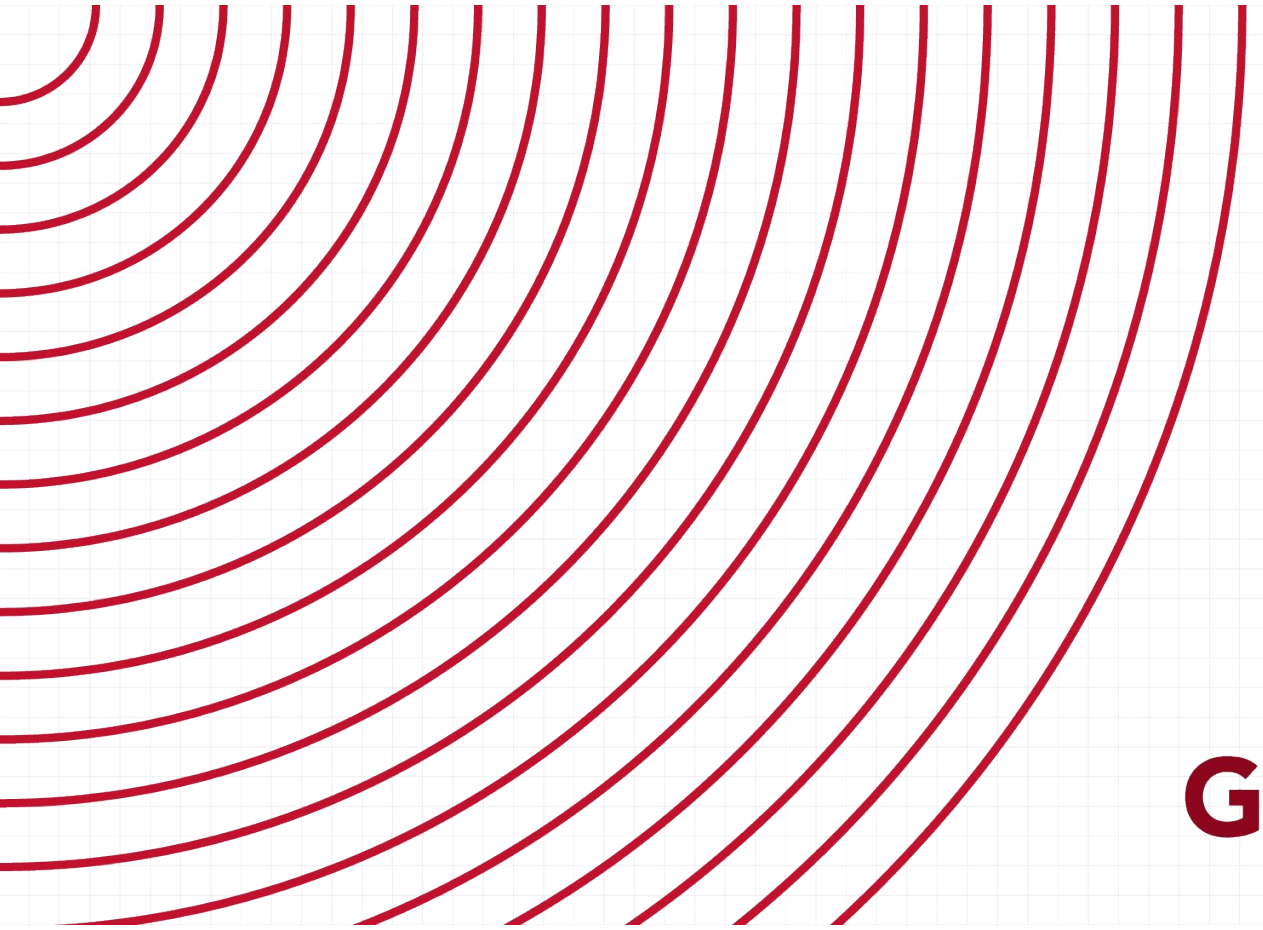
- El diseño y desarrollo de este curso aporta a la formación de profesionales en esta área.
- Es necesaria la correcta integración de recursos, actividades y roles para crear experiencias de aprendizaje exitosas
- La estructura de andamiaje es muy importante para permitir el progreso del estudiante
- La iteración del curso a partir de los comentarios de los estudiantes es fundamental para alcanzar experiencias de aprendizaje cada vez mejores

¿Cuáles son las contribuciones principales?

- Creación de la revisión del estado del arte de los cursos de computación natural
- Creación del diseño de aprendizaje de un curso introductorio de computación natural
- Versión piloto de los recursos de aprendizaje propuestos en el diseño
- Curaduría de videos y simuladores
- Despliegue de los recursos en ambiente virtual
- Opiniones de los estudiantes del curso creado

¿Cuáles son los trabajos futuros?

- Iterar nuevamente sobre los productos creados a partir de los comentarios recibidos
- Enriquecer el banco de retos
- Desarrollo de un simulador de algoritmos genéticos que cumpla con las expectativas planteadas
- Establecer un esquema para creación de nuevos laboratorios
- Evaluar el curso con más estudiantes
- Ofrecer el curso como una alternativa a la materia de CNYT en la escuela



ESCUELA
COLOMBIANA
DE INGENIERÍA
JULIO GARAVITO

Gracias

Índice de figuras

Índice de cuadros

Bibliografía

[1] DefiniciÓn de mitosis. https://www.enciclopediasalud.com/definiciones/mitosis.

[2] Meiosis. https://www.euston96.com/meiosis/.

[3] Multigrafos y grafos pesados. http://mate.cucei.udg.mx/matdis/5gra/5gra3.htm.

[4] Neurona definición, clases y tipo. https://www.educandose.com/neurona/.

[5] ¿qué es una mutaciÓn genÉtica? https://www.nascentis.com/genes.

[6] Mostafa Abdelraouf. Step-by-step artificial neural network visualizer. http://experiments.mostafa.io/public/ffbpann/. Accessed on 2021-09-20.

[7] D. Adams. *The Hitchhiker's Guide to the Galaxy*. San Val, 1995.

[8] Jesus albores. Neurona y aprendizaje. https://www.youtube.com/watch?v=82WmmM3yBho, 2019. Accessed on 2021-09-20.

[9] LB Almeida. C1. 2 multilayer perceptrons. *Handbook of Neural Computation*, 1997.

[10] James A. Anderson. *An Introduction to Neural Networks*. The MIT Press, 1995.

[11] Nelson Aparicio. simulacion seleccion natural. https://www.youtube.com/watch?v=kMryCfiJbGs, 2019. Accessed on 2021-10-04.

[12] Roohani B. Can ants choose the shortest path to a food source? https://www.youtube.com/watch?v=WekBF5kF5v4, 2020. Accessed on 2021-11-18.

[13] Janine Benyus. Biomimicry in action, 2009.

[14] biointeractive. Selección natural y adaptación | hhmi biointeractive video. https://www.youtube.com/watch?v=f98iDaryPj0, 2017. Accessed on 2021-10-04.

[15] Eric Bonabeau. From classical models of morphogenesis to agent-based models of pattern formation. July 1997.

[16] Anthony Brabazon, Michael O'Neill, and Sen McGarraghy. *Natural Computing Algorithms*. Springer Publishing Company, Incorporated, 1st edition, 2015.

[17] Kevin Brewer and Uri Wilensky. 2d genetic algorithm. http://modelingcommons.org/browse/one_model/4101#model_tabs_browse_nlw. Accessed on 2021-09-27.

[18] Scott Camazine, Nigel R. Franks, James Sneyd, Eric Bonabeau, Jean-Louis Deneubourg, and Guy Theraula. *Self-Organization in Biological Systems*. Princeton University Press, 2001.

[19] Fernando Sancho Caparrini. Cursos: Inteligencia artificial. http://www.cs.us.es/~fsancho/?p=inteligencia-artificial. Accessed on 2021-08-19.

[20] Fernando Sancho Caparrini. Multilayer perceptron. http://www.cs.us.es/~fsancho/Modelos/ANN.html. Accessed on 2021-09-20.

[21] L. Castro. Fundamentals of natural computing - basic concepts, algorithms, and applications. In *Chapman and Hall / CRC computer and information science series*, 2006.

[22] Kai-Chun Chu, Der-Juinn Horng, and Kuo-Chi Chang. Numerical optimization of the energy consumption for wireless sensor networks based on an improved ant colony algorithm. *IEEE Access*, 7:105562–105571, 2019.

[23] Kelly Cohen and Ali Minai. Teaching intelligent systems at the university of cincinnati. American Institute of Aeronautics and Astronautics, 3 2011.

[24] CuriosaMente. ¿por qué nos parecemos a nuestros papás? la genética - curiosamente 161. https://www.youtube.com/watch?v=axSh_Gl5GVo, 2019. Accessed on 2021-10-04.

[25] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems*, 2(4):303–314, December 1989.

[26] Victoria Dangeli. Explicación sobre el simulador de selección natural. https://www.youtube.com/watch?v=DRzwendMB_4, 2020. Accessed on 2021-10-04.

[27] Leandro De Castro, Yupanqui Muñoz, Leandro Freitas, and Charbel El-Hani. A virtual laboratory on natural computing: A learning experiment. *IJDET*, 6:55–73, 04 2008.

[28] Leandro Nunes de Castro. *Fundamentals of Natural Computing*. Chapman and Hall/CRC, June 2006.

[29] UPV/EHUko Kultura Zientifikoko Katedra Cátedra de Cultura Científica de la UPV/EHU. Ciencia express: selección natural. https://www.youtube.com/watch?v=Cz6VTtlQksE, 2014. Accessed on 2021-10-04.

[30] Instituto Nacional del Cáncer. gen. https://www.cancer.gov/espanol/publicaciones/diccionarios/diccionario-cancer/def/gen.

[31] J.L. Deneubourg and S. Goss. Collective patterns and decision-making. December 1989.

[32] Gordon Dent. Js aco. http://gordyd.github.io/js-aco/?problem=bayg29.tsp, 2017. Accessed on 2021-11-18.

[33] Rafael Dinis, Anabela Simões, and Jorge Bernardino. Graphea. ACM, 7 2013.

[34] Marco Dorigo and Thomas Stützle. The ant colony optimization metaheuristic: Algorithms, applications, and advances. 2003.

[35] D Dumitrescu. *Evolutionary computation*. CRC Press, Boca Raton, FL, 2000.

[36] Hortensia Díaz. How mendel´s pea plants helped us understand genetics. https://www.ted.com/talks/hortensia_jimenez_diaz_how_mendel_s_pea_plants_helped_us_understand_genetics#t-1463, 2013. Accessed on 2021-10-04.

[37] Esto EsAzio. ¿como encuentran tu comida las hormigas?|curiosidades. https://www.youtube.com/watch?v=TZstH3zz4Tc, 2019. Accessed on 2021-11-18.

[38] Tamandúa Estudio. ¿qué es la selección natural? - what is natural selection? https://www.youtube.com/watch?v=djLbljVY1QY, 2020. Accessed on 2021-10-04.

[39] Laurene Fausett. *Fundamentals of Neural Networks: Architectures, Algorithms, and Applications*. Prentice-Hall, Inc., 1994.

[40] Thiago Ferreira. Aco simulator. https://thiagodnf.github.io/aco-simulator/, 2021. Accessed on 2021-11-18.

[41] John Fulcher. Laboratory support for the teaching of neural networks. *The International Journal of Electrical Engineering & Education*, 35(1):29–36, 1998.

[42] GENIAL. Lo que verías si pudieras entrar a una colonia de hormigas. https://www.youtube.com/watch?v=6flnqhjiyJk, 2021. Accessed on 2021-11-18.

[43] Deborah M. Gordon. Ants at work - how an insect society is organized. 1999.

[44] Lisbetd Grajeda. Las teorías de darwin. https://www.youtube.com/watch?v=3WPFeGScj5M, 2016. Accessed on 2021-10-04.

[45] HarvardX. How a synapse works. https://www.youtube.com/watch?v=OvVl8rOEncE, 2017. Accessed on 2021-09-20.

[46] Simon Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 1999.

[47] LA VIDA DE LAS HORMIGAS. Como hablan las hormigas? reclutan para cazar - comunicacion y lenguaje feromonas pheidole pallidula. https://www.youtube.com/watch?v=GtxYRMHgQ6U, 2020. Accessed on 2021-11-18.

[48] Lee Jacobson. Ant colony optimization for hackers. https://www.theprojectspot.com/tutorial-post/ant-colony-optimization-for-hackers/10, 2015. Accessed on 2021-11-18.

[49] Colin G. Johnson. Teaching natural computation. *IEEE Computational Intelligence Magazine*, 4(1):24–30, 2009.

[50] Edward Keedwell and Soon-Thiam Khu. A hybrid genetic algorithm for the design of water distribution networks. *Engineering Applications of Artificial Intelligence*, 18(4):461–472, 2005.

[51] Donald Ervin Knuth. *The art of computer programming. 1. Fundamental algorithms.* Addison-Wesley, 1975.

[52] Jean-Baptiste Pierre Antoine de Monet de Lamarck. *Philosophie zoologique. vol. 1.* Paris :Duminil-Lesueur,.

[53] Pablo Lastra. Explicacion biológica genética de la evolución de las especies. https://www.youtube.com/watch?v=nweuEbxl-y0, 2015. Accessed on 2021-10-04.

[54] Cosimo Leuci. Minimal genetic algorithm. http://ccl.northwestern.edu/netlogo/models/community/Minimal%20Genetic%20Algorithm. Accessed on 2021-09-27.

[55] Jing Li and Wei Liu. An educational tool for the ant colony optimization algorithm. In *2009 First International Workshop on Education Technology and Computer Science*, volume 1, pages 55–58, 2009.

[56] Los Mascoteros. Ted: Dentro de la colonia de hormigas. https://www.youtube.com/watch?v=GHi1naxaFaA, 2016. Accessed on 2021-11-18.

[57] Rafael Matsunaga. Genetic cars. https://rednuht.org/genetic_cars_2/. Accessed on 2021-09-27.

[58] mdurance. potencial de acción. https://blog.mdurance.eu/academia/el-potencial-de-accion/.

[59] Gregor Mendel and Reginald Crundall Punnett and. *Versuche über Pflanzen-Hybriden /.* Im Verlage des Vereines„ 1866.

[60] Z. Michalewicz and M. Michalewicz. Teaching evolutionary algorithms. In *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, volume 3, pages 1702–1706 Vol. 3, 1999.

[61] Mark M. Millonas. Swarms, phase transitions, and collective intelligence. *arXiv: Adaptation and Self-Organizing Systems*, 1993.

[62] M. Mitchell, S. Tisue, and U. Wilensky. Netlogo robby the robot model. https://ccl.northwestern.edu/netlogo/models/RobbytheRobot, 2012. Accessed on 2021-09-27.

[63] Adham Atyabi Mohd Nadhir Ab Wahab, Samia Nefti-Meziani. Ant colony optimization algorithm processes. https://plos.figshare.com/articles/figure/_Ant_Colony_Optimization_Algorithm_processes_/1418788.

[64] Kumar Nishant, Pratik Sharma, Vishal Krishna, Chhavi Gupta, Kuwar Pratap Singh, Nitin, and Ravi Rastogi. Load balancing of nodes in cloud using ant colony optimization. In *2012 UKSim 14th International Conference on Computer Modelling and Simulation*, pages 3–8, 2012.

[65] THE UNIVERSITY of EDINBURGH. Undergraduate course: Natural computing (infr11161). http://www.drps.ed.ac.uk/18-19/dpt/cxinfr11161.htm. Accessed on 2021-08-19.

[66] Asociación Educar para el Desarrollo Humano. Aprendizaje, memoria y cerebro. https://www.youtube.com/watch?v=APU-9s-EMZU, 2012. Accessed on 2021-09-20.

[67] Asociación Educar para el Desarrollo Humano. Las neuronas. https://www.youtube.com/watch?v=zZ_1gLmtqJ4, 2013. Accessed on 2021-09-20.

[68] R. Poli. Evolutionary computation teaching at birmingham. IEEE.

[69] Tonis Pool. Visualisation of ant colony optimisation. https://courses.cs.ut.ee/demos/visual-aco/#/visualisation, 2015. Accessed on 2021-11-18.

[70] Educar Portal. Experiencia 360° | sinapsis, el cerebro por dentro. https://www.youtube.com/watch?v=SBia0Zv82VQ, 2019. Accessed on 2021-09-20.

[71] Primer. Simulando la selección natural. https://www.youtube.com/watch?v=0ZGbIKd0XrM, 2018. Accessed on 2021-10-04.

[72] Tran Vuong Quoc Anh. Neural network demo. https://lecture-demo.ira.uka.de/neural-network-demo/. Accessed on 2021-09-20.

[73] Dirk Riehle. Framework design: A role modeling approach. *Softwaretechnik-Trends*, 20, 2000.

[74] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, pages 65–386, 1958.

[75] Daniel Mariano Sanchez. Algoritmo de hormigas para encontrar el camino óptimo. 1/2. https://www.youtube.com/watch?v=yHPYSyw9kzY, 2020. Accessed on 2021-11-18.

[76] Daniel Mariano Sanchez. Algoritmo de hormigas para encontrar el camino óptimo. 2/2. https://www.youtube.com/watch?v=9QWroYC3nEg, 2020. Accessed on 2021-11-18.

[77] Daniel Smilkov and Shan Carter. Tensorflow playground. https://playground.tensorflow.org/. Accessed on 2021-09-20.

[78] A.E. Smith. Experiences with teaching adaptive optimization to engineering graduate students. IEEE.

[79] Dipti Srinivasan. Experience teaching a graduate level course in evolutionary computation. *IEEE Computational Intelligence Magazine*, 4, 8 2009.

[80] F. Stonedahl and U. Wilensky. Simple genetic algorithm. https://ccl.northwestern.edu/netlogo/models/SimpleGeneticAlgorithm, 2008. Accessed on 2021-09-27.

[81] Purdue University. Ie 590: Nature-inspired computing). https://engineering.purdue.edu/~mventresca/courses/IE590syllabusF16.pdf. Accessed on 2021-08-19.

[82] Anne Therese Venables and Grace Tan. A 'hands on' strategy for teaching genetic algorithms to undergraduates. *Journal of Information Technology Education: Research*, 6, 2007.

[83] Kangping Wang, Lan Huang, Chunguang Zhou, and Wei Pang. Particle swarm optimization for traveling salesman problem. In *2003 International Conference on Machine Learning and Cybernetics*, volume 3, pages 1583–1585. IEEE Press, November 2003.

[84] Karsten Weicker and Nicole Weicker. Teaching .evolutionary computation.at the university of stuttgart. 08 2001.

[85] Luis Estévez y Eva Salmerón BIOLOGíA Y GEOLOGíA. ¿cómo aprendemos? aprendizaje y conexiones neuronales. https://www.youtube.com/watch?v=j0p1CoKlamQ, 2019. Accessed on 2021-09-20.