Universidad Nacional
Federico Villarreal

Vicerrectorado de
INVESTIGACION

ESCUELA UNIVERSITARIA DE POSGRADO

"ANALIZADOR SINTÁCTICO DEL LENGUAJE NATURAL PARA EL IDIOMA CASTELLANO MEDIANTE REDES NEURONALES ARTIFICIALES"

TESIS PARA OPTAR EL GRADO ACADÉMICO DE

DOCTORA EN INGENIERÍA DE SISTEMAS

AUTOR:

Mg. MSc. SALLY KARINA TORRES ALVARADO

ASESOR:

DR. MAYHUASCA GUERRA JORGE VICTOR

JURADO:

DR. ANDRADE ARENAS LABERIANO MATÍAS

DR. LEZAMA GONZÁLEZ PEDRO MARTÍN

DR. RODRIGUEZ RODRIGUEZ CIRO

LIMA - PERÚ

2019

Índice **Pág.**

TITULO

“Analizador Sintáctico del Lenguaje Natural para el idioma Castellano Mediante Redes Neuronales Artificiales “.

AUTOR Mg. MSc. Sally Karina Torres Alvarado

ASESOR Dr. Jorge Víctor Mayhuasca Guerra.

RESUMEN (PALABRAS CLAVES)

La presente tesis presenta una propuesta de analizador sintáctico del lenguaje natural (en idioma castellano) aplicando redes neuronales artificiales utilizando oraciones escritas en idioma castellano. El análisis sintáctico del lenguaje Natural utilizando oraciones escitas, es abordada mediante la formalización de las estructuras sintácticas del lenguaje natural (Idioma castellano) los cuales tuvieron que formalizarse para los fines de lograr un reconocimiento de las estructuras sintácticas formalizadas.

En el presente trabajo se propone una solución al problema utilizando oraciones escritas, aplicando redes neuronales artificiales las cuales tras pasar por un proceso de aprendizaje con muestras reales proporcionadas palabras de una base de datos, logra realizar el reconocimiento de las estructuras sintácticas formalizadas. Los resultados obtenidos en un prototipo de una aplicación web, desarrollada en un software creado en PHP, con HTML, y java script sobre la plataforma de software libre Linux Fedora con Servidor web APACHE y gestor de Base de datos María DB, el cual me ha permitido demostrar que la propuesta logra realizar el reconocimiento de las estructuras sintácticas formalizadas del idioma castellano ingresadas a través de oraciones escritas.

Palabras clave: Lenguaje Natural; redes neuronales artificiales; analizador sintáctico, estructuras formalizadas.

ABSTRACT (KEY WORD)

In this thesis a proposal of syntactic analysis of natural language (in Spanish language) is presented, applying artificial neural networks using sentences written in Spanish. The syntactic analysis of natural language using written sentences, is addressed through the formalization of the syntactic structures of natural language (Spanish language) which had to be formalized for the purpose of achieving a recognition of formalized syntactic structures. In the present work a solution to the problem is proposed using written sentences, applying artificial neural networks which, after going through a learning process with real samples provided, words from a database, manages to realize the recognition of formalized syntactic structures. The results obtained in a prototype of a web application, developed in a software created in PHP, with HTML, and java script on the platform of free software Linux Fedora with Web Server APACHE and manager of Database Maria DB, which has allowed to demonstrate that the proposal manages to realize the recognition of the formalized syntactic structures of the Castilian language, entered through written sentences.

Key words: Natural Language; artificial neural networks; syntactic analyzer, formalized structures.

I. INTRODUCCION

El avance tecnológico ha permitido que se desarrollen sistemas con algunas características inteligentes por ello se requiere de tecnologías que permitan recibir las órdenes dadas por las personas y poder realizar una determinada acción. Un punto importante para la interacción con las maquinas es poder lograr la formalización del lenguaje natural para así lograr que las maquinas puedan entender las órdenes dadas por las personas, representando un problema en donde se debe buscar formas más eficientes de realizarlo. Algunas de las aplicaciones que requieren el uso del procesamiento del lenguaje natural son: Recuperación de información, Interfaces en lenguaje natural, Traducción automática, entre otros. La característica principal es que estas aplicaciones requieren una formalización del lenguaje natural.

En la actualidad existen sistemas que permiten resolver el problema de forma muy limitada en base a gramáticas de lenguajes formales, pero si se amplía

la cantidad de oraciones es necesario crear nuevas reglas gramaticales a diferencia de la propuesta con RNA que solo necesita ejecutar el proceso de aprendizaje de la red neuronal para que pueda hacer el reconocimiento de todas las combinaciones de oraciones formalizadas. Por otro lado, los sistemas podrían requerir aumentar el número de oraciones que entienden.

Por lo expuesto, en el presente trabajo se presenta una propuesta de análisis sintáctico del lenguaje natural aplicando redes neuronales artificiales usando archivos de texto idioma castellano. Para poder determinar si las palabras son parte de las estructuras formales a las que se llegó en la presente tesis y también si la oración corresponde a una de las estructuras sintácticas que se han formalizado. Por ello se planteó el problema de la siguiente forma ¿De qué manera se puede diseñar un analizador sintáctico para el

procesamiento del lenguaje natural para el idioma castellano? Para comprobarlo se implementó un SW, lográndose realizar el reconocimiento sintáctico de las oraciones.

1.1. PLANTEAMIENTO DEL PROBLEMA

Lo ideal sería que las computadoras y los sistemas computacionales entiendan un lenguaje natural hablado, por ello para describir los problemas de la presente tesis comenzaremos explicando la evolución del procesamiento en lenguaje natural, para luego analizar los avances que tienen los compiladores, y finalmente concluir en los problemas que involucra la creación de un analizador sintáctico de lenguaje natural para el idioma castellano.

1.-El procesamiento del lenguaje natural (PLN), es un área de investigación en esta en continuo desarrollo, se aplica actualmente en diferentes actividades como pueden ser la traducción automática, elaboración automática de resúmenes, sistemas de recuperación de información, interfaces en lenguaje natural, etc. Si bien es cierto que en los últimos años se han realizado muy buenos avances, los fundamentos teóricos del PLN se encuentran todavía en estado de desarrollo, por ello veamos la siguiente información histórica:

Entre los años de 1940-1960, se dieron las primeras aplicaciones del PLN las cuales tuvieron como interés fundamental la traducción automática. Los experimentos en esta área, se basaban en la substitución de palabra por palabra y se obtuvieron resultados rudimentarios.

Por tanto, surgió la necesidad de resolver problemas de ambigüedades que se presentaron tanto sintácticas como semánticas. Como por ejemplo la carencia de un orden de la estructura oracional en algunas lenguas, y la dificultad para obtener una representación tanto sintáctica como semántica. Cuando se lograron afrontar y en cierta

medida resolver se dio paso a una concepción más realista del lenguaje en la que era necesario contemplar las transformaciones que se producen en la estructura de la frase durante el proceso de traducción.

Por los años sesenta la comprensión del lenguaje fue el principal interés. Muchos de los trabajos estuvieron orientados a técnicas de análisis sintáctico. y por lo tanto la influencia de los trabajos en inteligencia artificial fue decisiva, centrándose principalmente en la representación del significado. Como resultado de estos trabajos se construyó el primer sistema de preguntas-respuestas el cual estaba basado en lenguaje natural. Como por ejemplo tenemos al proyecto *Eliza*, que reproducía las habilidades conversacionales de un psicólogo. Para realizarlo recogía patrones de información de las respuestas del cliente y elaboraba preguntas que simulaban una entrevista.

Entre los años 70 y 80, superados los problemas de construir programas más fiables. Aparecen un sin número de gramáticas que estaban orientadas a un tratamiento computacional, y se experimenta un notable crecimiento la tendencia hacia la programación lógica.

Por otro lado, en Europa surge un gran interés en la elaboración de programas que realizaban la traducción automática. Y se creó el proyecto de investigación *Eurotra*, el cual tenía como objetivo la traducción multilingüe. En Japón empiezan a aparecer equipos dedicados a la creación de productos de traducción para su distribución comercial.

Los últimos años se han caracterizado por la incorporación de técnicas estadísticas y se están desarrollando formalismos adecuados para el tratamiento de la información léxica. Se están introduciendo nuevas técnicas de representación del conocimiento

cercanas a la inteligencia artificial, y las técnicas de procesamiento utilizadas por investigadores procedentes del área de la lingüística e informática son cada vez más próximas. Surge intereses en la aplicación de estos avances en sistemas de recuperación de información con el objetivo de mejorar los resultados en consultas a texto completo.

En conclusión, es claro que los obstáculos a superar en el estudio del tratamiento del lenguaje son considerables, los resultados obtenidos y la evolución en los últimos años ubican al Procesamiento del lenguaje natural en una posición para poder dirigir las aplicaciones informáticas del futuro: los medios de comunicación del usuario con la computadora pueden ser más flexibles y el acceso a la información almacenada más eficiente.

Por ejemplo, con la creación de interfaces inteligentes el usuario tendría la facilidad para interactuar con la computadora en lenguaje natural. El uso de técnicas de PLN puede tener un alto impacto en la gestión documental y en los sistemas de traducción automática. Pero, por otro lado, la complejidad implícita en el tratamiento del lenguaje nos da limitaciones en los resultados y, por tanto, aplicaciones en áreas de conocimiento son muy concretas y con un uso restringido del lenguaje.

2.- Los diversos idiomas que se originaron en el mundo en las diversas naciones, también conocidos como lenguaje natural, son complejos y uno de los acercamientos más difundidos a un lenguaje natural son los lenguajes de programación, los cuales se crearon con el objetivo de comunicar a las computadoras o sistemas computaciones el orden de ejecución de instrucciones de un programa. Estos lenguajes de programación para lograr entender el lenguaje formal consta de dos fases frontend (las fases de análisis) y *back end*(las fases de generación y optimización de código). Estas fases se comunican a través de una codificación intermedia (generada por el *frontend*), que puede ser una

representación de la sintaxis del programa (un árbol sintáctico abstracto) o bien puede ser un programa en un lenguaje intermedio. El *frontend* depende del lenguaje fuente y casi siempre es independiente (o debe serlo) de la máquina objeto para la que se va a generar código; el *back end* depende del lenguaje objeto y debe ser independiente del lenguaje fuente (excepto quizá para algún tipo de optimización). El front end tiene tres etapas el análisis léxico, el análisis sintáctico y el análisis semántico. El back end tiene las siguientes etapas la generación del código intermedio, la optimización del código y por último la generación del código de máquina.

Como podemos ver una de las fases importantes en la interpretación de lenguajes, es el análisis sintáctico, el cual también es necesario realizar en el procesamiento del lenguaje natural, entonces desarrollar una forma eficiente de realizar el análisis sintáctico permitiría obtener una interface adecuada para las computadoras o sistemas computacionales. (Baliri, 2014)

3.-Una de las capacidades que tienen las personas para comunicarse, es el lenguaje el cual puede expresarse mediante signos orales o escritos.

El lenguaje natural es la forma que usamos las personas para comunicarnos con otras personas además el lenguaje puede realizarse mediante preguntas, expresar, emociones, describir hechos, etc, se trata entonces de un lenguaje aprendido como el español, el francés, el ruso, el inglés, etc. Realizar el análisis sintáctico para el lenguaje natural (castellano) es tedioso debido a los distintos casos gramaticales que se presentan y al número de palabras que tiene el lenguaje natural (castellano). También el proceso de realizar el análisis sintáctico requiere de la categorización de las estructuras gramaticales lo cual conlleva a un procesamiento muy complejo. Algunos de los métodos para resolver este problema, es utilizando teoría de lenguajes y compiladores, pero resultarían muy

difíciles de implementar por la cantidad de palabras del idioma castellano y porque las estructuras sintácticas son muy complejas en comparación de los lenguajes de programación.

1.2. DESCRIPCION DEL PROBLEMA

Existen dos tipos básicos y reconocidos de lenguajes: los lenguajes naturales y los lenguajes formales. El origen y desarrollo de los primeros, como pueden ser el castellano, el inglés o el francés, es natural, es decir, sin el control de ninguna teoría. Las teorías de lenguajes naturales y las gramáticas, fueron establecidas a priori, significa que fue después de que el lenguaje ya había madurado. De otro lado, los lenguajes formales como las matemáticas, la lógica, los lenguajes de programación, fueron desarrollados generalmente a través del establecimiento de una teoría, la cual le da las bases para dichos lenguajes. (Baliri, 2014).

Podemos definir el lenguaje como un conjunto de palabras. Cada lenguaje está compuesto por secuencias de símbolos tomados de alguna colección finita. En el caso de cualquier lengua natural (castellano, inglés, francés...), la colección finita es el conjunto de las letras del alfabeto junto con los símbolos que se usan para armar palabras (tales como el apóstrofe, el guion en el caso del inglés...).

El lenguaje castellano, extensivamente, puede ser definido como el conjunto (teóricamente infinito) de todas las oraciones en castellano. Como la mayoría de los lenguajes de interés, son recursivos en mayor o menor medida (es decir a partir de una oración, existen procedimientos que permiten formar otras mayores y más complejas oraciones), por ello debemos encontrar propiedades o conjuntos de propiedades, que las definan unívocamente (definición intensiva). Por ejemplo, dada la oración en castellano: “el auto es gris”, es posible construir otras oraciones como:

• mi amigo dice que la moto es roja

• si mi amigo dice que la moto es roja, es que la moto es roja

• si me contaron que mi amigo dice: que la moto es roja, entonces es cierto la moto es roja

Como es obvio, resultaría absurdo intentar escribir todas las posibles combinaciones de palabras que hay en el lenguaje castellano.

Por otro lado, el Procesamiento del Lenguaje Natural es una disciplina de la Inteligencia Artificial que se encarga de la formular e investigar sobre métodos de computación para mejorar la comunicación entre las personas y las maquinas mediante el uso de Lenguajes Naturales.

Los Lenguajes Naturales son los utilizados en la comunicación humana, ya sean escritos, hablados o signados una de las aplicaciones del procesamiento del lenguaje natural es su **comprensión.**

La sintaxis de otro lado, estudia los principios y los procesos por los cuales las oraciones son construidas en idiomas particulares como por ejemplo el castellano. Para un texto el análisis sintáctico es el proceso mediante el cual se pretende lograr la formación del árbol sintáctico, aplicando las reglas de producción de un lenguaje.

Por ello si queremos lograr comunicarnos con las maquinas en nuestro lenguaje natural (castellano) tenemos que resolver primero esta etapa, es decir la realización del análisis sintáctico ya que corresponde a una de las etapas de la construcción de compiladores de lenguajes. La realización del análisis sintáctico implica la revisión de las reglas de producción, las cuales definen un lenguaje para nuestro caso el castellano, y la verificación de la frase que se analiza para que cumpla con estas reglas de formación. De tal manera que si queremos resolver el procesamiento del lenguaje natural del Idioma

Castellano en formato escrito, como lo indica la teoría de lenguajes formales tendríamos que tomar en cuenta lo siguiente:

1. Una gramática libre de contexto,
2. El árbol sintáctico
3. La ambigüedad

1. Las gramáticas libres de contexto, también conocidas como gramáticas de tipo 2 o gramáticas independientes del contexto, son las que generan los lenguajes libres o independientes del contexto. Los lenguajes libres del contexto son aquellos que pueden ser identificados a través de un autómata de pila determinístico o no determinístico. Como toda gramática se definen mediante una cuádrupla:

$$G = (N, T, P, S),$$

Donde:

- N representa el conjunto finito de símbolos no terminales

- T representa el conjunto finito de símbolos terminales $N \cap T = \emptyset$

- P representa el conjunto finito de producciones

- S representa el símbolo distinguido o axioma $S \notin (N \cup T)$

En una gramática libre del contexto, cada producción de P tiene la forma

$$A \rightarrow \omega \qquad \{ A \in N \cup \{S\} \} \text{ o´ } A \rightarrow \omega \quad \{ \omega \in (N \cup T)^* - \{\varepsilon\} \}$$

2.- El Análisis sintáctico, consiste en determinar las funciones de las palabras o grupos de palabras dentro de la oración.

El árbol sintáctico es un grafo jerárquico con la propiedad de que cada uno de sus nodos representa un rol sintáctico. ¿El nodo raíz del árbol simboliza la unidad de expresión del

lenguaje, que para el caso del lenguaje natural equivale a la oración; De igual manera, las hojas del árbol representan las ocurrencias textuales de las palabras, y los nodos intermedios, los diferentes roles sintácticos presentes para cada palabra, junto con los que se pueden generar a partir de las reglas de producción definidas para el lenguaje (Chomsky, 1965). La forma como se conectan los nodos del árbol, está dada por un conjunto de reglas que definen qué secuencia de hijos puede formar un padre en un momento determinado. El objetivo de la realización del árbol sintáctico es, a partir de un conjunto de palabras, concluir si dicho conjunto está bien formado y equivale a una estructura que es reconocida para el lenguaje, el lenguaje natural en este caso. ¿Dentro de la generación de los diagramas, el analizador sintáctico ejecuta el primer paso del análisis al cumplir el papel de verificar si lo que el usuario ha ingresado, es una estructura acorde con el lenguaje natural; Además, define las diferentes partes que conforman el árbol, para su posterior análisis por las demás aplicaciones que así lo requieran.

3.- La ambigüedad en el contexto lingüístico es el fenómeno que ocurre cuando se puede obtener más de una interpretación a partir de una misma expresión dada (Bullinaria, 1997). Este fenómeno puede ser generado por la ambigüedad léxica (una palabra en un lenguaje determinado puede tener más de un rol sintáctico asociado) o por la ambigüedad estructural, (se puede representar una misma expresión con dos o más árboles sintácticos).

En conclusión, si consideramos la realización de este proceso por parte de un ser humano, podría implicar la mala interpretación de los elementos sintácticos del lenguaje, generando problemas de subjetividad e incompletitud que estarían reflejados a medida que incrementa la masa de los elementos a analizar.

Por ello las dificultades que tendríamos en el idioma castellano serian:

- Problemas (nivel morfológico): } Altamente flexivo: Múltiples procesos (flexión, derivación, composición) }
- No existen modelos morfológicos generales (muchas excepciones) }
- Número de palabras inmenso (decenas de millones) } 1,6-1,9 análisis por palabra (media) } Problemas (nivel sintáctico): }
- Carencia de estructura fija como en otros idiomas (ambigüedad)

Para la presente tesis se presentará una propuesta de cómo lograr este análisis con la posibilidad de sintetizar las reglas de producción de la parte sintáctica del Idioma castellano a través de un prototipo.

1.3. FORMULACION DEL PROBLEMA

1.3.1. PROBLEMA GENERAL

En base a lo expresado, el problema de la presente investigación se plantea de la siguiente forma:

¿Se puede diseñar un analizador sintáctico para el procesamiento del lenguaje natural para el idioma castellano?

1.3.2. PROBLEMAS ESPECIFICOS

- ¿Se puede realizar la identificación, categorización y codificación de las palabras del lenguaje Natural en idioma castellano?
- ¿Se puede realizar el análisis léxico del lenguaje natural en idioma castellano?
- ¿Se puede realizar la identificación, categorización y codificación de las estructuras sintácticas del lenguaje natural en idioma castellano? .

1.4. ANTECEDENTES

Mejorar la comunicación hombre maquina es el principal planteamiento que me motivo a plantear este tipo de investigación, puesto que los avances tecnológicos que tienen las máquinas están dirigidos a ello y una forma de lograrlo es utilizando el lenguaje natural porque de esta manera cualquier persona podría usar las máquinas.

1.4.1 INTERNACIONALES

Inicialmente el Procesamiento del Lenguaje Natural (PLN), empleo métodos que tuvieron gran aceptación y éxito, las dificultades surgieron, cuando sus aplicaciones se llevaron a la práctica, en entornos no controlados y utilizando vocabularios genéricos. Entre estas dificultades tenemos, por ejemplo, los problemas de polisemia y sinonimia. (Breál, 1897)

Problemática del procesamiento del Lenguaje Natural: la variación y la ambigüedad lingüísticas.-

El lenguaje natural, entendido como la herramienta que utilizan las personas para expresarse, posee propiedades que merman la efectividad de los sistemas de recuperación de información textual. Estas propiedades son la variación y la ambigüedad lingüística.

La variación lingüística se refiere a la posibilidad de utilizar diferentes palabras o expresiones para comunicar una misma idea. La variación lingüística provoca el silencio documental, es decir la omisión de documentos relevantes para cubrir la necesidad de información, ya que no se han utilizado los mismos términos que aparecen en el documento.

La ambigüedad lingüística se produce cuando una palabra o frase permite más de una interpretación. La ambigüedad implica el ruido documental, es decir la inclusión de

documentos que no son significativos, ya que se recuperan también documentos que utilizan el término, pero con significado diferente al requerido. Estas dos características dificultan considerablemente el tratamiento automatizado del lenguaje. (Vallez, 2007).

Para ilustrar mejor el problema algunos ejemplos de estos fenómenos en el proceso de recuperación de información:

- A nivel morfológico una misma palabra puede adoptar diferentes roles morfo-sintácticos en función del contexto en el que aparece, ocasionando problemas de ambigüedad (ejemplo 1).

Ejemplo 1.

Deja la comida que sobre sobre la mesa de la cocina, dijo llevando el sobre en la mano.

La palabra sobre es ambigua morfológicamente ya que puede ser un sustantivo masculino singular, una preposición, y también la primera o tercera persona del presente de subjuntivo del verbo sobrar.

- A nivel sintáctico, centrado en el estudio de las relaciones establecidas entre las palabras para formar unidades superiores, sintagmas y frases, se produce ambigüedad a consecuencia de la posibilidad de asociar a una frase más de una estructura sintáctica. Por otro lado, esta variación supone la posibilidad de expresar lo mismo, pero cambiando el orden de la estructura sintáctica de la frase. (ejemplo 2).

Ejemplo 2.

María vio a un niño con un telescopio en la ventana.

La interpretación de la dependencia de los dos sintagmas preposicionales, con un telescopio y en la ventana, otorga diferentes significados a la frase: (1) María vio a un niño que estaba en la ventana y que tenía un telescopio, (2) María estaba en la ventana, desde donde vio a un niño que tenía un telescopio, y (3) María estaba en la ventana, desde donde miraba con un telescopio, y vio a un niño.

- A nivel semántico, donde se estudia el significado de una palabra y el de una frase a partir de los significados de cada una de las palabras que la componen. La ambigüedad se produce porque una palabra puede tener uno o varios sentidos, es el caso conocido como polisemia (ejemplo 3).

Ejemplo 3.

Luís dejó el periódico en el banco.

El término banco puede tener dos significados en esta frase, (1) entidad bancaria y (2) asiento. La interpretación de esa frase va más allá del análisis de los componentes que forman la frase, se realiza a partir del contexto en que es formulada.

- Y también hay que tener en cuenta la variación léxica que hace referencia a la posibilidad de utilizar términos distintos a la hora de representar un mismo significado, es decir el fenómeno conocido como sinonimia (ejemplo 4):

Ejemplo 4:

Coche / Vehículo / Automóvil.

- A nivel pragmático, basado en la relación del lenguaje con el contexto en que es utilizado, en muchos casos no puede realizarse una interpretación literal y automatizada de los términos utilizados. En determinadas circunstancias, el

sentido de las palabras que forman una frase tiene que interpretarse a un nivel superior recurriendo al contexto en que es formulada la frase. (ejemplo 5).

Ejemplo 5.

Se moría de risa.

En esta frase no puede interpretarse literalmente el verbo morirse si no que debe entenderse en un sentido figurado.

- Otra cuestión de gran importancia es la ambigüedad provocada por la anáfora, es decir, por la presencia en la oración de pronombres y adverbios que hacen referencia a algo mencionado con anterioridad (ejemplo 6).

Ejemplo 6.

Ella le dijo que los pusiera debajo

La interpretación de esta frase tiene diferentes incógnitas ocasionadas por la utilización de pronombres y adverbio: ¿quién habló?, ¿a quién?, ¿qué pusiera qué?, ¿debajo de dónde? Por tanto, para otorgar un significado a esta frase debe recurrirse nuevamente al contexto en que es formulada.

Con estos ejemplos queda claro la complejidad del lenguaje y que su tratamiento automático no resulta fácil ni obvio. (Vallez, 2007)

Procesamiento lingüístico del lenguaje natural

Primero se identifican y analizan las palabras que forman un texto, luego se ve cómo éstas se relacionan y combinan entre sí para formar unidades superiores, los sintagmas y las frases. Por tanto, se trata de realizar el análisis sintáctico del texto. En este punto se aplican gramáticas (parsers) que son formalismos descriptivos del lenguaje que tienen por objetivo fijar la estructura sintáctica del texto (Sanderson, 2000). Las técnicas empleadas para aplicar y construir las gramáticas son muy variadas y

dependen del objetivo con el que se realiza el análisis sintáctico. En el caso de la recuperación de la información acostumbra a aplicarse un análisis superficial, donde se identifican únicamente las estructuras más significativas: frases nominales, sintagmas verbales y preposicionales, entidades, etc. Este nivel de análisis suele utilizarse para optimizar recursos y no ralentizar el tiempo de respuesta de los sistemas.

A partir de la estructura sintáctica del texto, el siguiente objetivo es obtener el significado de las frases que lo componen. Se trata de conseguir la representación semántica de las frases, a partir de los elementos que la forman. (Vallez, 2007)

Análisis sintáctico. -

Por otro lado, el Análisis sintáctico tiene como función etiquetar cada uno de los componentes sintácticos que aparecen en la oración y analizar cómo las palabras se combinan para formar construcciones gramaticalmente correctas. El resultado de este proceso consiste en generar la estructura correspondiente a las categorías sintácticas formadas por cada una de las unidades léxicas que aparecen en la oración.

Las gramáticas, tal como se muestra en la siguiente figura, están formadas por un conjunto de reglas:

O --> SN, SV

SN --> Det, N

SN --> Nombre Propio

SV --> V, SN

SV --> V

SP --> Preposición, SN

SN = sintagma nominal

SV = sintagma verbal

Det = determinante (Sosa, 1997)

Ejemplo de una gramática simple: las reglas tienen como función la composición de estructuras (figura 1.1)

El resultado del análisis se puede expresar en forma arbórea. Los árboles son formas gráficas utilizadas para expresar la estructura de la oración, consistentes en nodos etiquetados (O, SN, SV..) conectados por ramas:

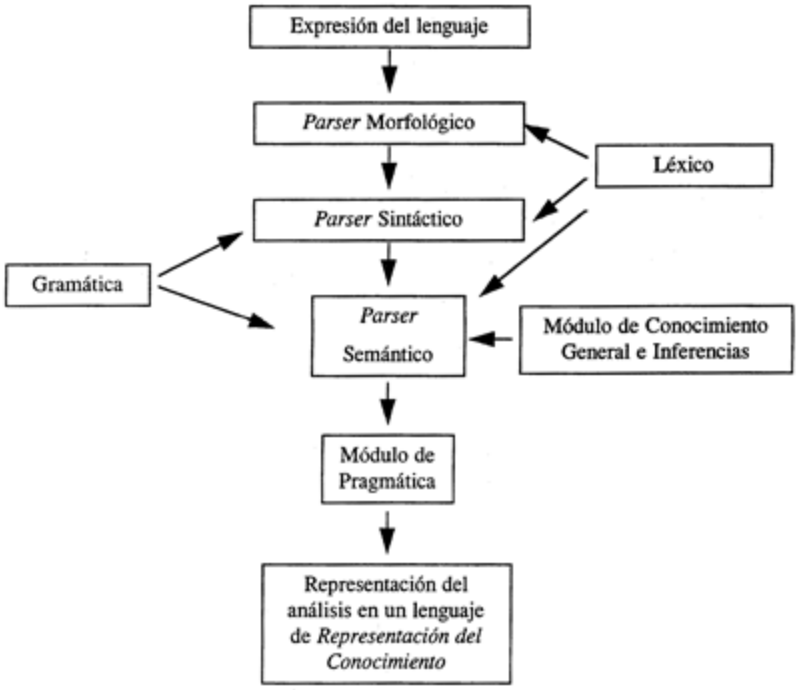


Figura 1.1 Composición de estructuras para gramáticas (Sosa, 1997)

Si bien esta propuesta utilizando reglas para procesamiento de lenguaje natural tiene un resultado adecuado, no logra implementar el lenguaje en su totalidad debido a que sería necesario definir toda la gramática del idioma castellano a través de reglas formalizadas.

1.4.2 NACIONALES

A nivel nacional no existen trabajos relevantes en el campo del procesamiento de lenguaje natural para el idioma castellano.

1.5. JUSTIFICACION DE LA INVESTIGACION

1.5.1. TEÓRICA

En este contexto la presente investigación hace un gran aporte al conocimiento científico y tecnológico ya que permitirá ser usado como base para sistemas que necesiten procesar y entender sentencias en lenguaje natural del idioma castellano. Por lo que la presente investigación tiene una importancia y queda justificada.

1.5.2. PRÁCTICA

En la presente investigación se realizará una aplicación práctica en el prototipo que se desarrollará para realizar las pruebas de análisis sintáctico al lenguaje natural en castellano y puede ser usado para crear futuras interfaces de nuevos sistemas operativos de computadoras y dispositivos móviles. Por lo que plenamente justificado la investigación.

1.5.3. METODOLÓGICA

En la presente investigación se realizará aplicando los principios metodológicos adecuados que permitan encontrar una propuesta que resuelva el problema planteado por lo que queda justificado.

1.5.4. SOCIAL

En la presente investigación se propondrá una solución que no afecte en el aspecto social a la sociedad y más bien provee de una herramienta que ayuda a la interacción con la computadora y dispositivos móviles por lo que queda socialmente justificada.

1.6. LIMITACIONES DE LA INVESTIGACION

Existen diversas formas de realizar análisis sintáctico, los cuales se utilizan en compiladores de lenguajes de programación sin embargo está demostrado que funcionan de manera eficiente en lenguajes formales, pero el uso de ellos en el procesamiento del lenguaje natural no ha tenido buenos resultados.

La presente Investigación estará limitada por el número fijo de estructuras sintácticas que tiene el Idioma castellano, para de esta manera poder probar el con la ayuda de la red neuronal un sinnúmero de oraciones del idioma castellano en formato escrito que tendrá que ser analizado por un prototipo de laboratorio de analizador sintáctico, pero que permita realizar los experimentos correspondientes en su totalidad.

1.6.1. ESPACIAL

La presente tesis está dirigida a todos los países de habla castellana en América tenemos sur América y centro América, en Europa España, en África Guinea Ecuatorial y en países como Filipinas, Marruecos, etc. es uno de los dos o tres idiomas oficiales.

1.6.2. TEMPORAL

La presente tesis corresponde a una investigación actual ya que el avance de las comunicaciones con las maquinas es algo inminente ahora usamos teclados, pero en un futuro próximo será hablado y por lo tanto esta tesis ayudará a mejorar esta comunicación hombre máquina. La tesis iniciara cuando sea aprobado el plan de tesis y se tiene proyectado concluir en un periodo de un año.

1.6.3. SOCIAL

Socialmente está delimitado a todas las personas de habla hispana de los cinco continentes, ya que en la actualidad el uso de computadoras y dispositivos móviles está generalizado, y como la presente tesis pretende proporcionar una teoría que ayude a mejorar la interacción hombre máquina.

1.7. OBJETIVOS

1.7.1. OBJETIVO GENERAL

Desarrollar un analizador sintáctico de lenguaje natural para el idioma castellano

1.7.2. OBJETIVOS ESPECIFICOS

- Determinar la identificación, categorización y codificación de las palabras del lenguaje Natural en idioma castellano.
- Realizar el análisis léxico del lenguaje natural en idioma castellano.
- Realizar la identificación, categorización y codificación de las estructuras sintácticas del lenguaje natural en idioma castellano

1.7.3. HIPOTESIS

- HIPOTESIS GENERAL

Con el uso una red neuronal artificial se podría realizar el análisis sintáctico del lenguaje natural para el idioma castellano.

- HIPOTESIS ESPECIFICAS
  - Es posible realizar la identificación, categorización y codificación de las palabras del Idioma castellano mediante la formalización de las palabras para su procesamiento a nivel computacional,

- Es posible realizar el análisis léxico del lenguaje natural para el idioma castellano mediante una red neuronal artificial.
- Es posible realizar la identificación, categorización y codificación de las estructuras sintácticas del lenguaje natural para el Idioma castellano mediante una red neuronal artificial.

II. MARCO TEORICO

Los sistemas de computación secuencial, son exitosos en la resolución de problemas matemáticos o científicos, en la creación, manipulación y mantenimiento de bases de datos, en comunicaciones electrónicas, en el procesamiento de textos, gráficos y auto edición, incluso en funciones de control de electrodomésticos, haciéndolos más eficientes y fáciles de usar, pero definitivamente tienen una gran incapacidad para interpretar el mundo.

Esta dificultad de los sistemas de cómputo que trabajan bajo la filosofía de los sistemas secuenciales, desarrollados por Von Neuman, ha hecho que un gran número de investigadores centre su atención en el desarrollo de nuevos sistemas de tratamiento de la información, que permitan solucionar problemas cotidianos, tal como lo hace el cerebro humano; este órgano biológico cuenta con varias características deseables para cualquier sistema de procesamiento digital, tales como:

- Es robusto y tolerante a fallas, diariamente mueren neuronas sin afectar su desempeño.
- Es flexible, se ajusta a nuevos ambientes por aprendizaje, no hay que programarlo.
- Puede manejar información difusa, con ruido o inconsistente.
- Es altamente paralelo
- Es pequeño, compacto y consume poca energía.

El cerebro humano constituye una computadora muy notable, es capaz de interpretar información imprecisa suministrada por los sentidos a un ritmo increíblemente veloz. Logra discernir un susurro en una sala ruidosa, un rostro en un callejón mal iluminado y leer entre líneas un discurso; lo más impresionante de todo, es que el cerebro aprende sin

instrucciones explícitas de ninguna clase, a crear las representaciones internas que hacen posibles estas habilidades.

Basados en la eficiencia de los procesos llevados a cabo por el cerebro, e inspirados en su funcionamiento, varios investigadores han desarrollado desde hace más de 30 años la teoría de las Redes Neuronales Artificiales (RNA), las cuales emulan las redes neuronales biológicas, y que se han utilizado para aprender estrategias de solución basadas en ejemplos de comportamiento típico de patrones; estos sistemas no requieren que la tarea a ejecutar se programe, ellos generalizan y aprenden de la experiencia.

La teoría de las RNA ha brindado una alternativa a la computación clásica, para aquellos problemas, en los cuales los métodos tradicionales no han entregado resultados muy convincentes, o poco convenientes. Las aplicaciones más exitosas de las RNA son:

1. Procesamiento de imágenes y de voz
2. Reconocimiento de patrones
3. Planeamiento
4. Interfaces adaptivas para sistemas Hombre/máquina
5. Predicción
6. Control y optimización
7. Filtrado de señales

Los sistemas de cómputo tradicional procesan la información en forma secuencial; un computador serial consiste por lo general de un solo procesador que puede manipular instrucciones y datos que se localizan en la memoria, el procesador lee, y ejecuta una a una las instrucciones en la memoria; este sistema serial es secuencial, todo sucede en una sola secuencia determinística de operaciones. Las RNA no ejecutan instrucciones, responden en paralelo a las entradas que se les presenta. El resultado no se almacena en

una posición de memoria, este es el estado de la red para el cual se logra equilibrio. El conocimiento de una red neuronal no se almacena en instrucciones, el poder de la red está en su topología y en los valores de las conexiones (pesos) entre neuronas.

Las RNA son una teoría que aún está en proceso de desarrollo, su verdadera potencialidad no se ha alcanzado todavía; aunque los investigadores han desarrollado potentes algoritmos de aprendizaje de gran valor práctico, las representaciones y procedimientos de que se sirve el cerebro, son aún desconocidas. Tarde o temprano los estudios computacionales del aprendizaje con RNA acabarán por converger a los métodos descubiertos por evolución, cuando eso suceda, muchos datos empíricos concernientes al cerebro comenzarán súbitamente a adquirir sentido y se tornarán factibles muchas aplicaciones desconocidas de las redes neuronales. (Vemuri, 1990)

Lenguajes Compiladores. -

Es una rama de la informática que se encarga del diseño, implementación, análisis caracterización y clasificación de los lenguajes de programación.

El proceso de compilación es una secuencia de varias fases. Como ya lo habíamos dicho en post anteriores, cada fase toma su información de entrada de la fase anterior, y cada fase tiene su propia interpretación del código fuente. Para ilustrar mejor las fases del compilador veamos el siguiente diagrama de bloques:

figura 2.1.1

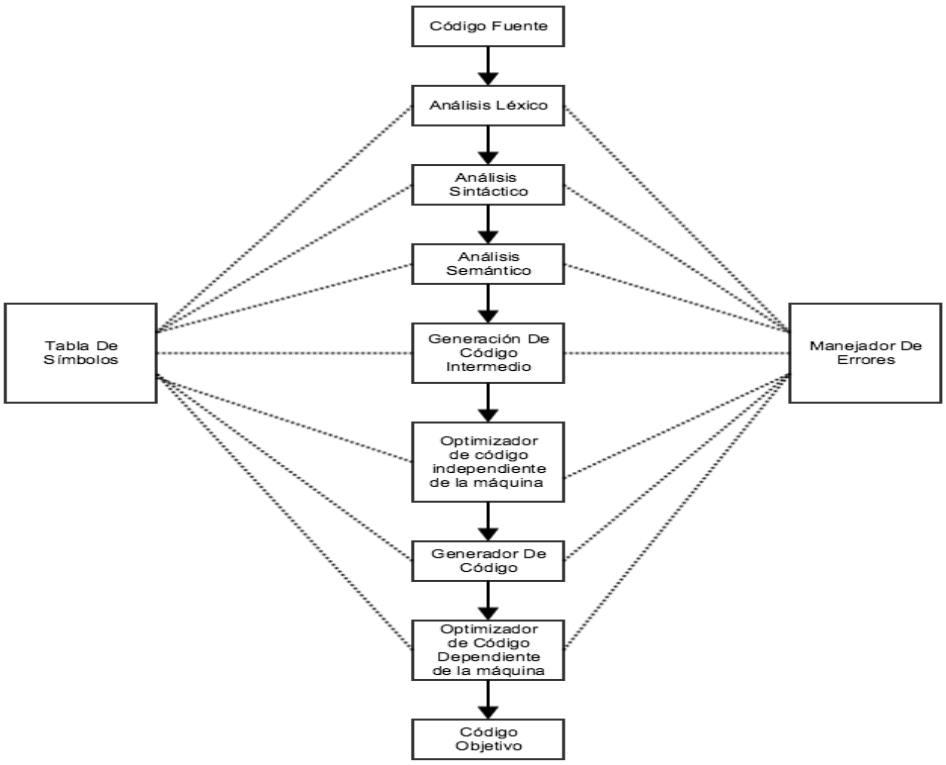


Figura 2.1.1 Diagrama de bloques de un compilador

Introducimos entonces los conceptos básicos del proceso de compilación que en post futuros veremos más a detalle con ejemplos prácticos, usando distintos tipos de tecnología. (Aho,Alfred; Ulman Jeffrey, 2004)

Análisis Léxico. -

Así como todos los idiomas tiene un alfabeto de caracteres válidos (e.g: español, chino, japonés), cada lenguaje tiene su propio conjunto de caracteres aceptados. Algunos lenguajes tienen un alfabeto muy reducido como brainfuck que consta de solo 8 caracteres, hasta los más extensos y conocidos como java, python o javascript.

Ahora bien, la fase de análisis léxico se encarga de convertir el código fuente del usuario en un *stream* de caracteres que al final se convierten en un conjunto de lexemas con significados específicos que denominamos *tokens.*

Análisis Sintáctico

Esta fase que le sigue al análisis léxico, la llamamos Análisis sintáctico o en ingles *parsing* o como muchos programadores le dicen *parseo* (aunque según la RAE no existe tal vocablo… aun) En esta fase se toma el conjunto de tokens producidos por la fase de análisis léxico y se genera un árbol de sintaxis, luego este se revisa de acuerdo a la gramática formal del lenguaje definido y se verifica que las expresiones construidas por el arreglo de tokens sean sintácticamente correctos.

Análisis Semántico. -

El análisis semántico valido si el árbol sintáctico construido concuerda con las reglas del lenguaje formal. Por ejemplo, asignaciones de valores entre tipos de datos que son compatibles. En esta fase también es muy necesario mantener un control de los identificadores con sus respectivos tipos y expresiones, por ejemplo, si una variable es asignada sin haber sido declarada, y produce como salida un árbol de sintaxis anotado.

Generación de código intermedio. -

Después de haber conducido el análisis semántico el compilador genera un código intermedio entre el código fuente y el código de la maquina objetivo (unos y ceros). Este representa un programa para una máquina abstracta. Está en medio de un lenguaje de alto nivel y un lenguaje de máquina. Este código intermedio debe ser generado de tal manera que es fácilmente traducido a un lenguaje de máquina de bajo nivel.

Optimización. -

La siguiente fase realiza una optimización del código intermedio generado. La optimización puede asumirse como algo que quita líneas de código innecesarias, y ordena

una secuencia de declaraciones que aceleran la ejecución del programa sin desperdiciar recursos de CPU o memoria RAM.

Por ejemplo, si tenemos las siguientes líneas en código intermedio.

- **int valorA=0;**
- **int valor=10;**
- **int temp=valora+valor;**

El optimizador lograría que este código se viera reducido así

- **int temp=0 +10;**

Reduciendo así el espacio reservado para las dos variables iniciales y los ciclos de reloj que toma asignarlos.

Generación De Código

En la fase de generación de código se toma la versión optimizada del código intermedio y se mapea al lenguaje de máquina objetivo. La generación de código traduce entonces el código intermedio en una secuencia reubicable de código de máquina, más tarde el enlazador del sistema operativo tomara estas instrucciones y les asignara un espacio en la memoria para así poder funcionar.

Tabla de símbolos. -

Es importante entender que la tabla de símbolos no es más que una estructura de datos que denominamos matriz ortogonal, que es una tabla con posiciones de memoria dinámicas sobre las cuales se puede realizar búsquedas y editar los valores de cada una de sus celdas.

El compilador se vale de esta estructura de datos para almacenar identificadores, tipos y valores de las variables a lo largo de todo el proceso de compilación, de forma que sea

fácil y rápido obtener cualquiera de las propiedades de un símbolo. (Aho,Alfred; Ulman Jeffrey, 2004).

2.1 MARCO CONCEPTUAL

2.1.1 REDES NEURONALES ARTIFICIALES

Funcionamiento de una neurona Biológica

El cerebro humano tiene un gran número de neuronas (aproximadamente 10^{11}) que se encuentran altamente interconectados (aproximadamente 10^4 conexiones por elemento). Las neuronas se componen de: dendritas, cuerpo de la célula o soma, y axón. Las dendritas, vienen a ser el árbol receptor de la red, son como fibras nerviosas que cargan de señales eléctricas el cuerpo de la célula. El cuerpo de la célula, realiza la suma de esas señales de entrada. El axón es una fibra que mide mas de un metro para alcanzar las extremidades y transmite un impulso eléctrico a las dendritas de otras neuronas. La unión entre un axón de una célula y una dendrita de otra célula es llamado sinapsis, la longitud de la sinapsis es determinada por la complejidad del proceso químico que estabiliza la función de la red neuronal (Vemuri, 1990). Un esquema simplificado de la interconexión de dos neuronas biológicas se observa en la figura 2.1.2

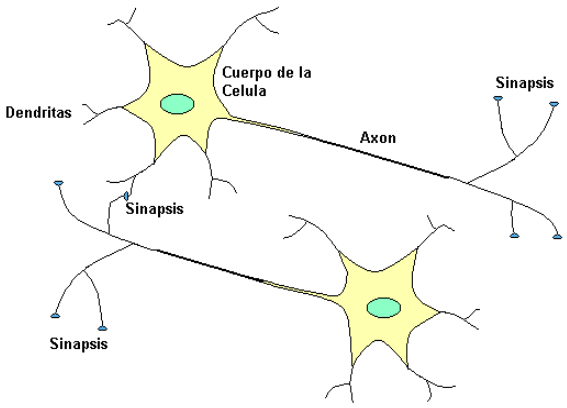


Figura 2.1.2 Neuronas Biológicas

Algunas de las estructuras neuronales son determinadas en el nacimiento, otra parte es desarrollada a través del aprendizaje, proceso en que nuevas conexiones neuronales son realizadas y otras se pierden por completo.

Las estructuras neuronales continúan cambiando durante toda la vida, estos cambios consisten en el refuerzo o debilitamiento de las uniones sinápticas; por ejemplo, se cree que nuevas memorias son formadas por la modificación de esta intensidad entre sinapsis, así el proceso de recordar el rostro de un nuevo amigo, consiste en alterar varias sinapsis.

Como consecuencia de los primeros estudios sobre la base neural de los sistemas mnémicos (relacionados con la memoria), se creía que el almacenamiento de la memoria asociativa, tanto implícita como explícita, requerían de un circuito neuronal muy complejo. Entre quienes comenzaron a oponerse a este enfoque se hallaba Donald O. Hebb, profesor de la universidad de Milner; Hebb sugirió que el aprendizaje asociativo podría ser producido por un mecanismo celular sencillo y propuso que las asociaciones podrían formarse por una actividad neuronal coincidente: "Cuando un axón de la célula A excita la célula B y participa en su activación, se produce algún proceso de desarrollo o cambio metabólico en una o en ambas células, de suerte que la eficacia de A, como célula excitadora de B, se intensifica". Según la regla Hebbiana de aprendizaje, el que coincida la actividad de las neuronas presinápticas (suministran el impulso de entrada) con la de las postsinápticas (reciben el impulso) es muy importante para que se refuerce la conexión entre ellas, este mecanismo es llamado pre-postasociativo, del cual puede observarse un ejemplo en la figura 2.1.3 (Vemuri, 1990)

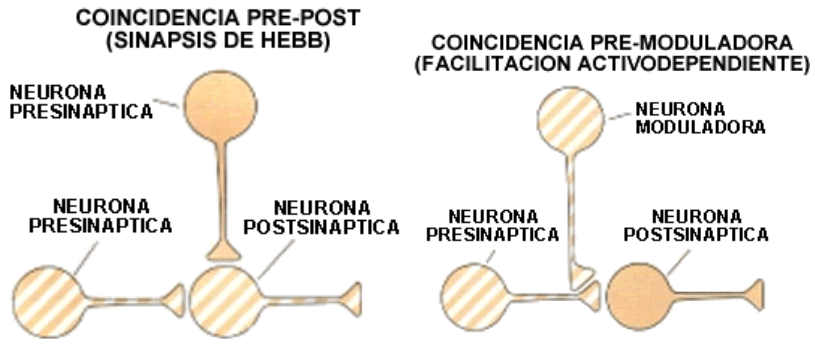


Figura 2.1.3 Cambios asociativos de las fuerzas sinápticas durante el aprendizaje

Todas las neuronas conducen la información de forma similar, esta viaja a lo largo de axones en breves impulsos eléctricos, denominados potenciales de acción; los potenciales de acción que alcanzan una amplitud máxima de unos 100 mV y duran 1 ms, son resultado del desplazamiento a través de la membrana celular de iones de sodio dotados de carga positiva, que pasan desde el fluido extracelular hasta el citoplasma intracelular; la concentración extracelular de sodio supera enormemente la concentración intracelular.

La membrana en reposo mantiene un gradiente de potencial eléctrico de -70mv, el signo negativo se debe a que el citoplasma intracelular está cargado negativamente con respecto al exterior; los iones de sodio no atraviesan con facilidad la membrana en reposo, los estímulos físicos o químicos que reducen la gradiente de potencial, o que despolaricen la membrana, aumentan su permeabilidad al sodio y el flujo de este ion hacia el exterior acentúa la despolarización de la membrana, con lo que la permeabilidad al sodio se incrementa más aún. (Vemuri, 1990)

Alcanzado un potencial crítico denominado "umbral", la realimentación positiva produce un efecto regenerativo que obliga al potencial de membrana a cambiar de signo. Es decir, el interior de la célula se torna positivo con respecto al exterior, al cabo de 1 ms, la permeabilidad del sodio decae y el potencial de membrana retorna a -70mv,

su valor de reposo. Tras cada explosión de actividad iónica, el mecanismo de permeabilidad del sodio se mantiene refractario durante algunos milisegundos; la tasa de generación de potenciales de acción queda así limitada a unos 200 impulsos por segundo, o menos.

Aunque los axones puedan parecer hilos conductores aislados, no conducen los impulsos eléctricos de igual forma, como hilos eléctricos no serían muy valiosos, pues su resistencia a lo largo del eje es demasiado grande y a resistencia de la membrana demasiado baja; la carga positiva inyectada en el axón durante el potencial de acción queda disipada uno o dos milímetros más adelante, para que la señal recorra varios centímetros es preciso regenerar frecuentemente el potencial de acción a lo largo del camino la necesidad de reforzar repetidamente esta corriente eléctrica limita a unos 100 metros por segundo la velocidad máxima de viaje de los impulsos, tal velocidad es inferior a la millonésima de la velocidad de una señal eléctrica por un hilo de cobre.

Los potenciales de acción, son señales de baja frecuencia conducidas en forma muy lenta, estos no pueden saltar de una célula a otra, la comunicación entre neuronas viene siempre mediada por transmisores químicos que son liberados en las sinápsis. Un ejemplo de comunicación entre neuronas y del proceso químico de la liberación de neurotransmisores se ilustra en la figura 2.1.4.

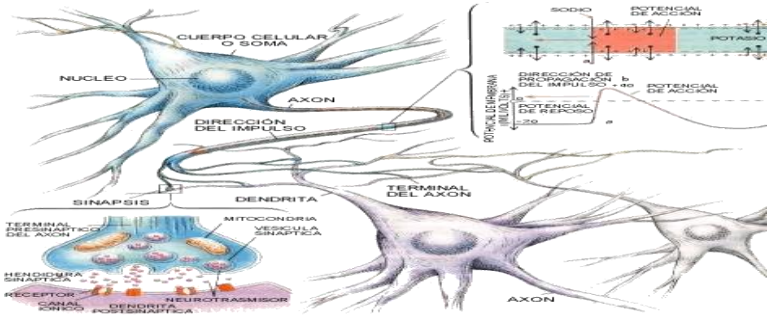


Figura 2.1.4 Comunicación entre neuronas

Cuando un potencial de acción llega al terminal de un axón son liberados transmisores alojados en diminutas vesículas, que después son vertidos en una hendidura de unos 20 nanómetros de anchura que separa la membrana presináptica de la postsináptica; durante el apogeo del potencial de acción, penetran iones de calcio en el terminal nervioso, su movimiento constituye la señal determinante de la exocitosis sincronizada, esto es la liberación coordinada de moléculas neurotransmisoras. En cuanto son liberados, los neurotransmisores se enlazan con receptores postsinápticos, instando el cambio de la permeabilidad de la membrana.

Cuando el desplazamiento de carga hace que la membrana se aproxime al umbral de generación de potenciales de acción, se produce un efecto excitador y cuando la membrana resulta estabilizada en la vecindad el valor de reposo se produce un efecto inhibidor. Cada sinápsis produce sólo un pequeño efecto, para determinar la intensidad (frecuencia de los potenciales de acción) de la respuesta cada neurona ha de integrar continuamente hasta unas 1000 señales sinápticas, que se suman en el soma o cuerpo de la célula.

En algunas neuronas los impulsos se inician en la unión entre el axón y el soma, y luego se transmiten a lo largo del axón a otras células nerviosas. El axón se divive en varias ramificaciones si esta cerca de otra célula, formando la sinápsis con el axón de otra célula. Dependiendo del neurotransmisor que se libere, la sinápsis puede ser excitatorias o inhibitorias aproximadamente a cada neurona le llega de 10.000 a 100.000 sinápsis y su axón realiza una cantidad igual de sinápsis.

Las sinápsis son de tres tipos y se clasifican según su posición en la superficie de la neurona receptora: axo-somática, axo-dendrítica, axo-axónica. Los fenómenos que

ocurren en la sinápsis son de naturaleza química, pero tienen efectos eléctricos laterales que se pueden medir. (Vemuri, 1990)

En la figura 2.1.5 se visualiza el proceso químico de una sinápsis y los diferentes elementos que hacen parte del proceso tanto en la neurona presináptica, como en la postsináptica.

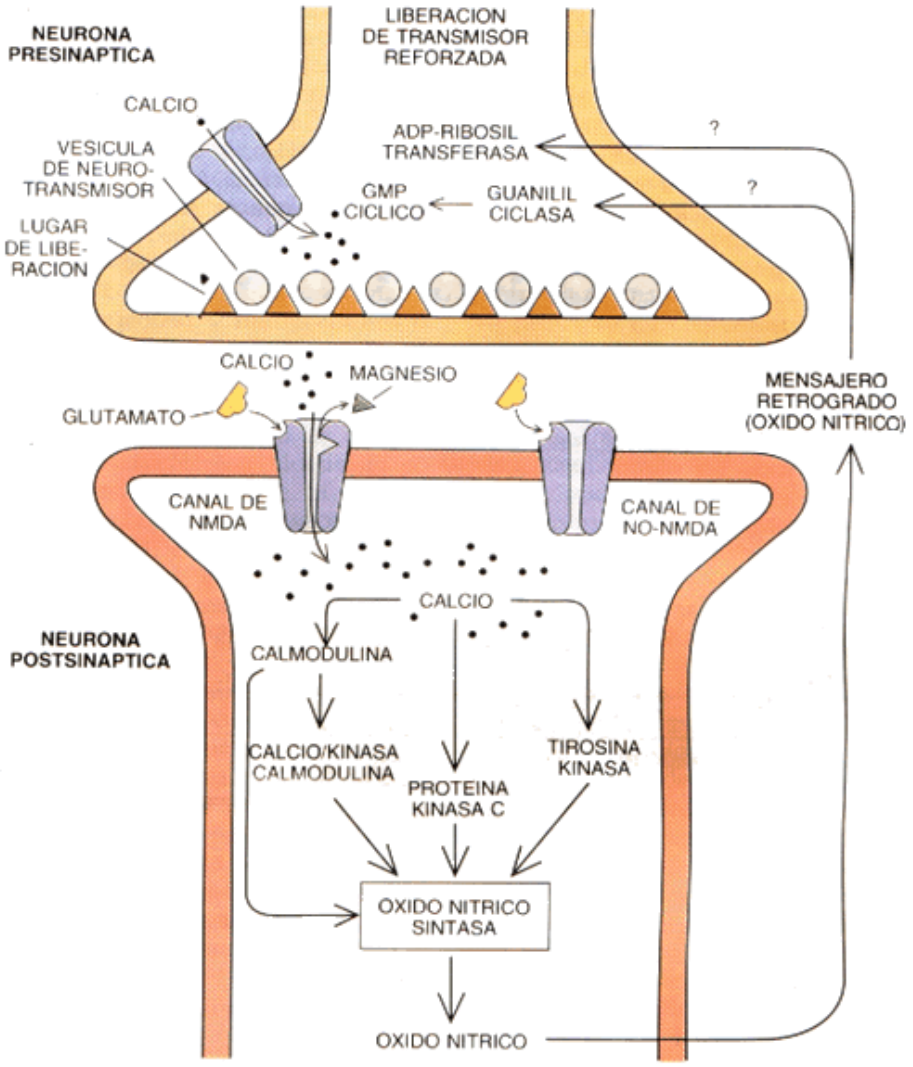


Figura 2.1.5 Proceso químico de una sinápsis

Las RNA no es mucho más sencilla que el cerebro, pero, hay dos aspectos semejantes entre las redes biológicas y las redes artificiales, primero los bloques de construcción

de ambas redes son elementos computacionales sencillos conectados entre si; segundo, estas conexiones entre neuronas definen la función de la red. (Vemuri, 1990).

Características de una red Neuronal Artificial

La neurona artificial también es conocida como nodo, neuronodo o elemento de procesamiento; la neurona artificial y la neurona biológica son similares como se muestra En la figura 2.1.6.

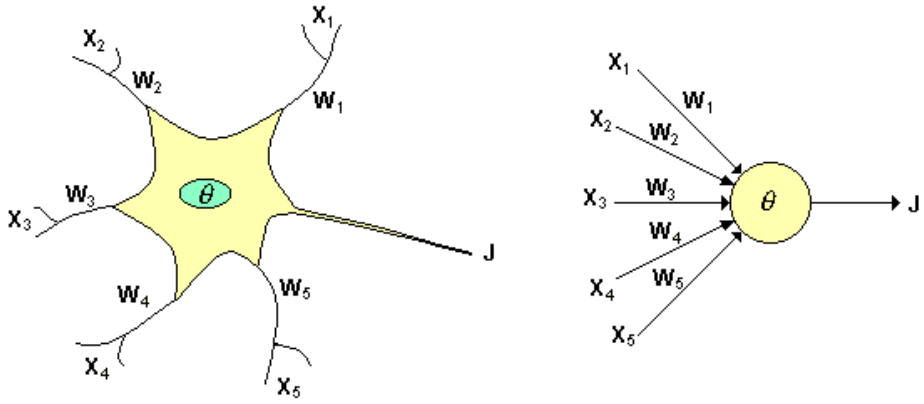


Figura 2.1.6 De la neurona biológica a la neurona artificial

Existen algunas analogías entre la neurona biológica y la neurona artificial como son:

- las señales que llegan desde otras neuronas son equivalentes a las dendritas y se representan como el vector de entradas X_i.
- Dos neuronas se conectan mediante la intensidad de la sinápsis que son los pesos de cada entrada representado con el vector W_i el cual tiene valores reales.
- Para que se active una neurona debe sobrepasar el umbral ϑ , similar a lo que ocurre en el cuerpo de la celula biologica.

Las entrada en una neurona artificial son continuas a diferencia de las neuronas biológica. Cada señal de entrada pasa a través de una ganancia o peso, llamado peso sináptico o fortaleza de la conexión cuya función es análoga a la de la función sináptica

de la neurona biológica. Los pesos pueden ser positivos (excitatorios), o negativos (inhibitorios), el nodo sumatorio acumula todas las señales de entradas multiplicadas por los pesos o ponderadas y las pasa a la salida a través de una función umbral o función de transferencia. La entrada neta a cada unidad puede escribirse según la fórmula 2.1.7

$$neta_i = \sum_{i=1}^{n} W_i X_i = \vec{X}\,\vec{W} \qquad (2.1.7)$$

Una idea clara de este proceso se muestra en la figura 2.1.8, en donde puede observarse el recorrido de un conjunto de señales que entran a la red.

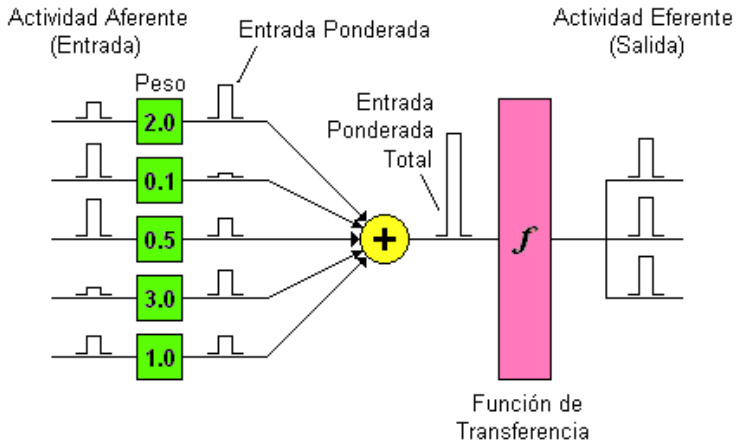


Figura 2.1.8 Proceso de una red neuronal

Una vez que se ha calculado la activación del nodo, el valor de salida fórmula 2.1.9, equivale a

$$x_i = f_i(neta_i) \qquad (2.1.9)$$

Donde f_i representa la función de activación para esa unidad, que corresponde a la función escogida para transformar la entrada $neta_i$ en el valor de salida x_i y que depende de las características específicas de cada red.

Notación: Una notación matemática estándar no ha sido aún establecida para las redes neuronales, ya que sus aplicaciones son útiles en muchos campos, Ingeniería, Física, Sicología y Matemáticas. En este trabajo se adoptó la siguiente convención para identificar las variables, de manera que fuera compatibles con las diferentes áreas, siendo lo más sencilla posible:

- Valores escalares: se representarán por medio de letra minúscula itálica
- Vectores: se representarán con letra itálica minúscula en negrilla.
- Matrices: se representarán con letra mayúscula itálica en negrilla.

Para redes multicapa, los parámetros se adoptarán según la siguiente forma 2.1.10:

$$W^{C}_{S^C,S^C} \quad (2.1.10)$$

Donde c, es el número de la capa a la que corresponde dicho peso, y s representa las neuronas que participan en proceso.

Así $W^{2}_{1,1}$ representa el peso de la segunda capa que comunica la primera neurona de dicha capa con la primera neurona de la primera capa. De igual manera el peso que representa la conexión desde la última neurona de la capa dos a la última neurona de la capa uno estará representado por la fórmula 2.1.11:

$$W^{2}_{S^2,S^1} \quad (2.1.11)$$

Esta convención es adoptada para todos los parámetros de la red.

Funciones de Transferencia: Un modelo más académico que facilita el estudio de una neurona, puede visualizarse en la figura 2.1.12

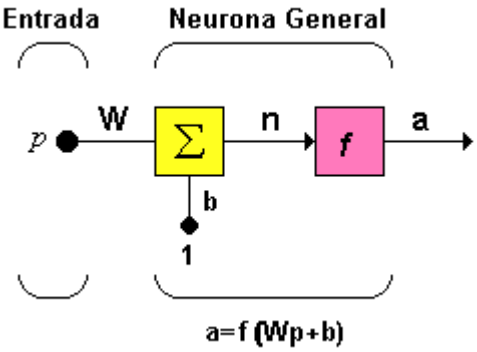


Figura 2.1.12 Neurona de una sola entrada

Las entradas a la red serán ahora presentadas en el vector p, que para el caso de una sola neurona contiene solo un elemento, w sigue representando los pesos y la nueva entrada b es una ganancia que refuerza la salida del sumador n, la cual es la salida neta de la red; la salida total está determinada por la función de transferencia , la cual puede ser una función lineal o no lineal de n, y que es escogida dependiendo de las especificaciones del problema que la neurona tenga que resolver; aunque las RNA se inspiren en modelos biológicos no existe ninguna limitación para realizar modificaciones en las funciones de salida, así que se encontrarán modelos artificiales que nada tienen que ver con las características del sistema biológico.

Limitador fuerte (Hardlim): Es una función de transferencia que si el valor de la red es menor a 0 la salida es 0 y si es mayor o igual a 0 es 1 como se ve en la figura 2.1.13, y se muestra en la fórmula 2.1.14.

$$a = \begin{Bmatrix} 1 & si & n \geq 0 \\ 0 & si & n < 0 \end{Bmatrix} \quad (2.1.14)$$

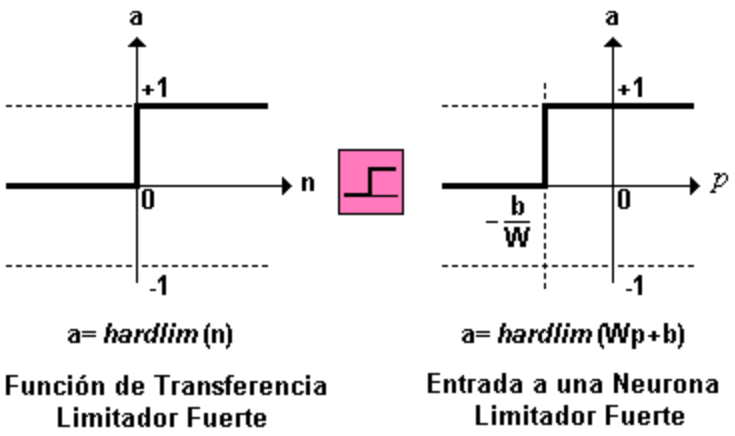


Figura 2.1.13 Función de transferencia Hardlim

El icono para la función Hardlim reemplazara a la letra *f* en la expresión general, cuando se utilice la función Hardlim.

Una modificación de esta función puede verse en la figura 2.1.15, la que representa la función de transferencia Hardlims que restringe el espacio de salida a valores entre 1 y –1 (2.1.16)

$$a = \begin{cases} 1 & si \quad n \geq 0 \\ -1 & si \quad n < 0 \end{cases} \quad (2.2.15)$$

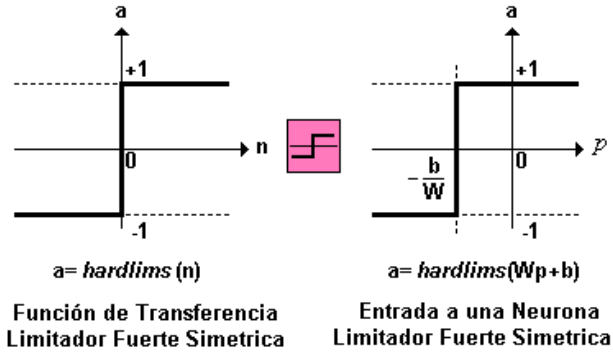


Figura 2.1.15 Función de transferencia Hardlims

Función de transferencia lineal (purelin): esta función de transferencia lineal tiene una salida igual como se ve en la figura 2.1.17 y en la fórmula 2.1.18.

$$a = n \quad (2.1.18)$$

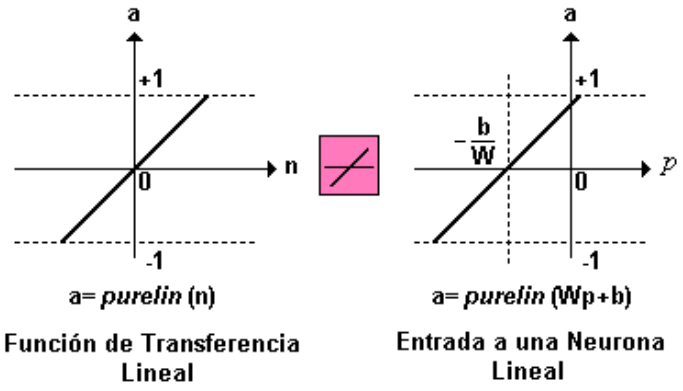


Figura 2.1.17 Función de transferencia lineal

En la gráfica del lado derecho de la figura 2.1.17, puede verse la característica de la salida *a* de la red, comparada con la entrada *p*, más un valor de ganancia *b*, neuronas que emplean esta función de transferencia son utilizadas en la red tipo Adaline.

Función de transferencia sigmoidal (logsig): Esta función toma los valores entre mas y menos infinito como se muestra en la figura 2.1.19 y da una salida entre 0 y 1, de acuerdo a la expresión 2.1.20

$$a = \frac{1}{1 + e^{-n}} \quad (2.1.20)$$

Esta función es comúnmente usada en redes multicapa, como la Backpropagation, en parte porque la función logsig es diferenciable.

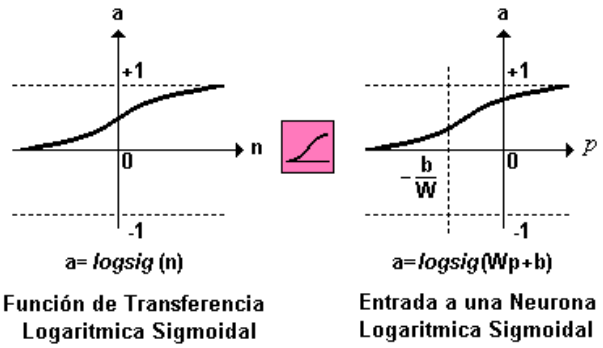


Figura 2.1.19 Función de transferencia sigmoidal

La tabla 2.1.1 hace una relación de las principales funciones de transferencia empleadas en el entrenamiento de redes neuronales.

Nombre	**Relación Entrada /Salida**	**Icono**	**Función**
Limitador Fuerte	$a = 0 \; n < 0$ $a = 1 \; n \geq 0$		*hardlim*
Limitador Fuerte Simétrico	$a = -1 \; n < 0$ $a = +1 \; n \geq 0$		*hardlims*
Lineal Positiva	$a = 0 \; n < 0$ $a = n \; 0 \leq n$		*poslin*
Lineal	$a = n$		*purelin*
Lineal Saturado	$a = 0 \; n < 0$ $a = n \; 0 \leq n \leq 1$ $a = 1 \; n > 1$		*satlin*
Lineal Saturado Simétrico	$a = -1 \; n < -1$ $a = n \quad -1 \leq n \leq 1$ $a = +1 \; n > 1$		*satlins*
Sigmoidal Logarítmico	$a = \frac{1}{1 + e^{-n}}$		*logsig*
Tangente Sigmoidal Hiperbólica	$a = \frac{e^{n} - e^{-n}}{e^{n} + e^{-n}}$		*tansig*

Competitiva	$a = 1$ *Neurona con n max* $a = 0$ *El resto de neuronas*	C	*compet*

Tabla 2.1.1 Funciones de Transferencia

Topología de una Red: Típicamente una neurona tiene más de una entrada; en la figura 2.1.21 se observa una neurona con R entradas; las entradas individuales $p_1, p_2, ..., p_R$ son multiplicadas por los pesos correspondientes $w_{1,1}$, $w_{1,2}$,...$w_{1,R}$ pertenecientes a la matriz de pesos ***W***.

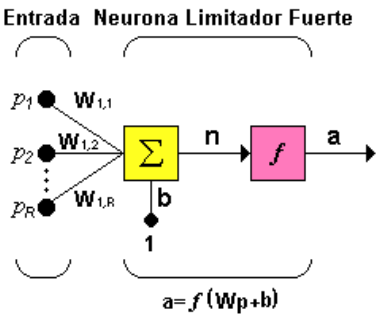


Figura 2.1.21 Neurona con múltiples entradas

La neurona tiene una ganancia *b*, la cual llega al mismo sumador (2.1.22) al que llegan las entradas multiplicadas por los pesos, para formar la salida *n*,

$$n = w_{1,1}p_1 + w_{1,2}p_2 + \ldots + w_{1,R}p_R + b \quad (2.1.22)$$

Esta expresión puede ser escrita en forma matricial (2.1.23)

$$n = \boldsymbol{Wp} + b \quad (2.1.23)$$

Los subíndices de la matriz de pesos representan los términos involucrados en la conexión, el primer subíndice representa la neurona destino y el segundo, representa la fuente de la señal que alimenta a la neurona. Por ejemplo, los índices de $w_{1,2}$indican que este peso es la conexión desde la segunda entrada a la primera neurona. Esta convención se hace más útil cuando hay más de una neurona, o cuando se tiene una

neurona con demasiados parámetros; en este caso la notación de la figura 2.1.23, puede resultar inapropiada y se prefiere emplear la notación abreviada representada en la figura 2.1.24

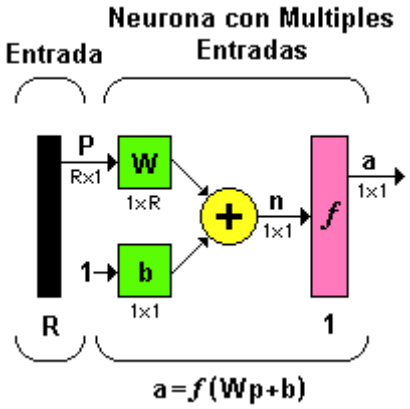


Figura 2.1.24 Neurona con múltiples entradas, notación abreviada

El vector de entrada ***p*** es representado por la barra sólida vertical a la izquierda. Las dimensiones de ***p*** son mostradas en la parte inferior de la variable como Rx1, indicando que el vector de entrada es un vector fila de R elementos. Las entradas van a la matriz de pesos ***W***, la cual tiene R columnas y solo una fila para el caso de una sola neurona. Una constante 1 entra a la neurona multiplicada por la ganancia escalar *b*. La salida de la red *a*, es en este caso un escalar, si la red tuviera más de una neurona ***a*** sería un vector.

La red neuronal esta conformada por capas que tienen un conjunto de neuronas y de acuerdo a la ubicación son: capa de entrada, capa oculta y capa de salida.

Por la Capa de entrada ingresan las señales a la red y no se realizan cálculos.

Las Capas ocultas no tienen contacto con el medio exterior, y son estas las que determinan las diferentes topologías de la red.

La capa de salida recibe la información de la capa oculta y transmite la respuesta al medio externo.

Una red de una sola capa con un número *S* de neuronas, se observa en la figura 2.1.25 en la cual, cada una de las R entradas es conectada a cada una de las neuronas, la matriz de pesos tiene ahora *S* filas.

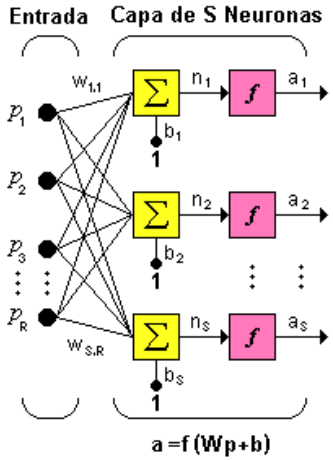


Figura 2.1.25 Capa de *S* neuronas

La capa incluye la matriz de pesos, los sumadores, el vector de ganancias, la función de transferencia y el vector de salida. Esta misma capa se observa en notación abreviada en la figura 2.1.26

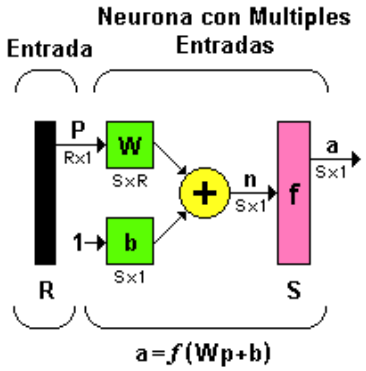


Figura 2.1.26 Capa de S neuronas con notación abreviada

En la figura 2.1.26 se han dispuesto los símbolos de las variables de tal manera que describan las características de cada una de ellas, por ejemplo la entrada a la red es el vector **p** cuya longitud R aparece en su parte inferior, **W** es la matriz de pesos con

dimensiones SxR expresadas debajo del símbolo que la representa dentro de la red, **a** y **b** son vectores de longitud S el cual, como se ha dicho anteriormente representa el número de neuronas de la red.

Ahora, si se considera una red con varias capas, o red multicapa, cada capa tendrá su propia matriz de peso **W**, su propio vector de ganancias **b**, un vector de entradas netas **n**, y un vector de salida **a**. La versión completa y la versión en notación abreviada de una red de tres capas, pueden ser visualizadas en las figuras 2.1.27 y 2.1.28, respectivamente.

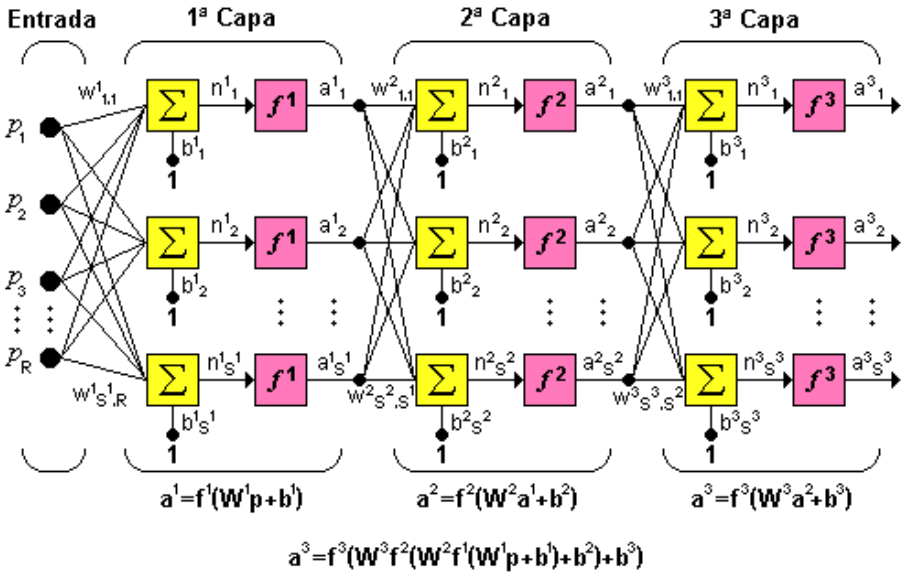


Figura 2.1.27 Red de tres capas

Para esta red se tienen R entradas, S^1 neuronas en la primera capa, S^2 neuronas en la segunda capa, las cuales pueden ser diferentes; las salidas de las capas 1 y 2 son las entradas a las capas 2 y 3 respectivamente, así la capa 2 puede ser vista como una red de una capa con $R=S^1$ entradas, $S^1=S^2$ neuronas y una matriz de pesos $\boldsymbol{W^2}$de dimensiones S^1xS^2

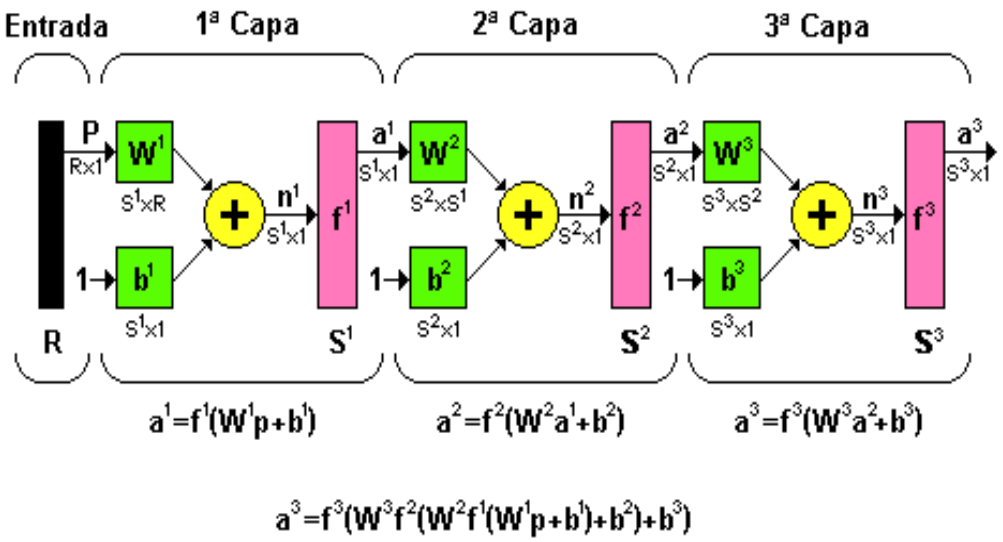


Figura 2.1.28 Red de tres capas con notación abreviada

Las redes multicapa son más poderosas que las redes de una sola capa, por ejemplo, una red de dos capas que tenga una función sigmoidal en la primera capa y una función lineal en la segunda, puede ser entrenada para aproximar muchas funciones de forma aceptable, una red de una sola capa no podría hacer esto como se verá en capítulos posteriores.

Un tipo de redes, un poco diferente a las que se han estudiado hasta el momento, son las redes recurrentes, estas contienen una realimentación hacia atrás o retroalimentación, es decir algunas de sus salidas son conectadas a sus entradas. Un tipo de red recurrente de tiempo discreto es mostrado en la figura 2.1.29.

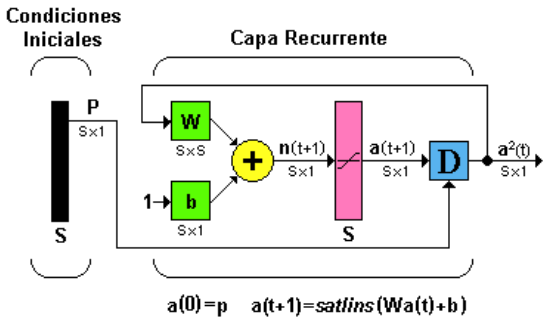


Figura 2.1.29 Redes Recurrentes

Para este tipo particular de red el vector ***p*** suple las condiciones iniciales (***a***(0)= ***p***), y la salida está determinada por (2.1.30):

$$\boldsymbol{a}(1) = satlins(\boldsymbol{W}\boldsymbol{a}(0) + \boldsymbol{b}), \quad \boldsymbol{a}(2) = satlins(\boldsymbol{W}\boldsymbol{a}(1) + \boldsymbol{b}) \quad (2.1.30)$$

Donde a(1) y a(2), corresponden a la salida de la red para el primer y segundo intervalo de tiempo, respectivamente. La red alcanzará su estado estable cuando la salida para un instante de tiempo sea la misma salida del instante de tiempo anterior.

Las redes recurrentes son potencialmente más poderosas que las redes con realimentación hacia delante. En este tipo de redes se introducen también dos nuevos conceptos, el bloque de retardo de la figura 2.2.30 y el bloque integrador de la figura 2.1.33.

- Retardo

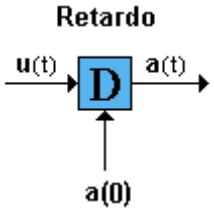


Figura 2.1.31 Bloque de retardo

$$\boldsymbol{a}(t) = \boldsymbol{u}(t-1) \quad (2.1.32)$$

La salida del bloque de retardo es el valor de entrada retrasado en un paso de tiempo, este bloque requiere que la salida sea inicializada con el valor a(0) para el tiempo t=0; a(0) se convierte en la salida de la red para el instante de tiempo inicial.

- Integrador

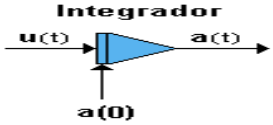


Figura 2.1.33 Bloque integrador

La salida del integrador es calculada de acuerdo a la expresión (2.1.34)

$$\boldsymbol{a}(t) = \int_0^t \boldsymbol{u}(\tau) d\tau + \boldsymbol{a}(0) \qquad (2.1.34)$$

En general las redes neuronales se pueden clasificar de diversas maneras, según su topología, forma de aprendizaje (supervisado o no supervisado), tipos de funciones de activación, valores de entrada (binarios o continuos); un resumen de esta clasificación se observa en la figura 2.1.35

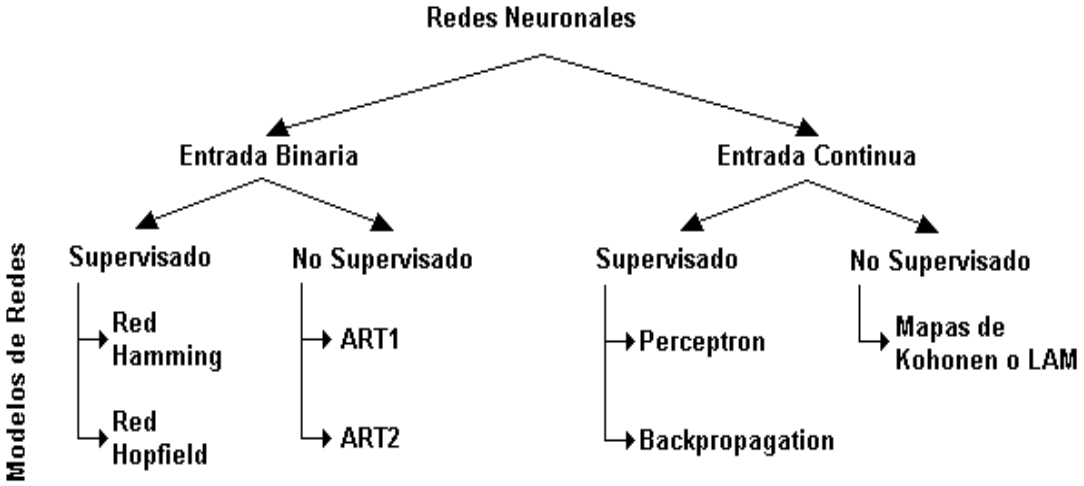


Figura 2.1.35 Clasificación de las Redes Neuronales

Perceptrón

Antecedentes: La primera red neuronal conocida, fue desarrollada en 1943 por Warren McCulloch y Walter Pitts; esta consistía en una suma de las señales de entrada, multiplicadas por unos valores de pesos escogidos aleatoriamente. La red tipo Perceptrón fue inventada por el sicólogo Frank Rosenblatt en el año 1957. Su intención era ilustrar algunas propiedades fundamentales de los sistemas inteligentes en general, sin entrar en mayores detalles con respecto a condiciones específicas y desconocidas para organismos biológicos concretos.

Estructura de la red:

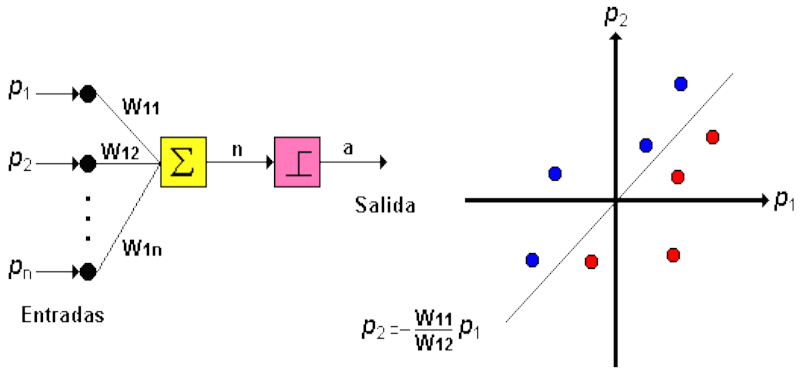


Figura 2.1.36 Perceptrón

El perceptron calcula la suma n del producto de las entradas por los pesos y pasa el resultado a la función de transferencia tipo escalón. Si el patrón es de la clase A se aactiva la salida (+1)y si es de la clase B se desactica (-1) como se ve en la figura 2.1.36. La red tipo Perceptrón emplea principalmente dos funciones de transferencia, *hardlim* con salidas 1, 0 o *hardlims* con salidas 1, -1; su uso depende del valor de salida que se espera para la red, es decir si la salida de la red es unipolar o bipolar; sin embargo la función *hardlims* es preferida sobre la *hardlim*, ya que el tener un cero multiplicando algunas de los valores resultantes del producto de las entradas por el vector de pesos, ocasiona que estos no se actualicen y que el aprendizaje sea más lento.

Una técnica utilizada para analizar el comportamiento de redes como el Perceptrón es presentar en un mapa las regiones de decisión creadas en el espacio multidimensional de entradas de la red, en estas regiones se visualiza qué patrones pertenecen a una clase y cuáles a otra, el Perceptrón separa las regiones por un hiperplano cuya ecuación queda determinada por los pesos de las conexiones y el valor umbral de la función de activación de la neurona, en este caso los valores de los pesos pueden fijarse o adaptarse empleando diferentes algoritmos de entrenamiento. (Vemuri, 1990)

Para ilustrar el proceso computacional del Perceptrón consideremos la matriz de pesos (2.1.37) en forma general.

$$W = \begin{bmatrix} W_{1,1} & W_{1,2} & \cdots & W_{1.R} \\ W_{2,1} & W_{2,2} & \cdots & W_{2,R} \\ W_{S,1} & W_{S,2} & \cdots & W_{S,R} \end{bmatrix} \quad (2.1.37)$$

Los pesos para una neurona están representados por un vector (2.1.38) compuesto de los elementos de la *i*-ésima fila de ***W***

$$w = \begin{bmatrix} w_{i,1} \\ w_{i,2} \\ \vdots \\ w_{i,R} \end{bmatrix} \quad (2.1.38)$$

De esta forma y empleando la función de transferencia hardlim(2.1.39)la salida de la neurona *i* de la capa de salida

$$a_i = hardlim\left(n_i\right) = hardlim\left(w_i{}^T p_i\right) \quad (2.1.39)$$

El Perceptrón es una neurona artificial, con una capa de entrada y una de salida, por lo cual solo puede discriminar patrones sencillos linealmente separables, el caso más conocido es la imposibilidad del Perceptrón de representar la función OR EXCLUSIVA.

Regla de aprendizaje: El Perceptrón es un tipo de red de aprendizaje supervisado, es decir necesita conocer los valores esperados para cada una de las entradas presentadas; su comportamiento está definido por pares (2.1.40) de esta forma:

$$\{p_1, t_1\}, \{p_2, t_2\}, \ldots, \{p_Q, t_Q\} \quad (2.1.40)$$

Cuando *p* es aplicado a la red, la salida de la red es comparada con el valor esperado *t*, y la salida de la red está determinada (2.1.41) por:

$$a = f\left(\sum_i w_i p_i\right) = hardlims\left(\sum_i w_i p_i\right) \quad (2.1.41)$$

Los valores de los pesos determinan el funcionamiento de la red, estos valores se pueden fijar o adoptar utilizando diferentes algoritmos de entrenamiento de la red.

En el proceso de entrenamiento el Perceptrón se expone a un conjunto de patrones de entrada y los pesos de la red son ajustados de forma que al final de entrenamiento se obtengan las salidas esperadas para cada uno de esos patrones de entrada.

El algoritmo de entrenamiento del Perceptrón puede resumirse en los siguientes pasos:

1. Se inicializa la matriz de pesos y el valor de la ganancia, por lo general se asignan valores aleatorios a cada uno de los pesos w_i y al valor *b.*

2. Se presenta el primer patrón a la red, junto con la salida esperada en forma de pares entrada/salida

3. Se calcula la salida de la red por medio de (2.1.42)

$$a = f(w_1 p_1 + w_2 p_2 + b) \quad (2.1.42)$$

Donde *f* puede ser la función *hardlim* o *hardlims*

4. Cuando la red no retorna la salida correcta, es necesario alterar el valor de los pesos, tratando de llevarlo hasta *p* y así aumentar las posibilidades de que la clasificación sea correcta, una posibilidad es adicionar *p* a *w* haciendo que el vector *w* apunte en la dirección de *p*, y de esta forma después de repetidas presentaciones de *p*a la red, *w* se aproximará asintóticamente a *p*; este es el procedimiento adoptado para la regla de aprendizaje del Perceptrón.

El proceso de aprendizaje del Perceptrón puede definirse en tres reglas, las cuales cubren la totalidad de combinaciones de salidas y sus correspondientes valores esperados. Estas reglas utilizando la función de transferencia *hardlim*, se expresan (2.1.43, 2.1.44, 2.1.45) como sigue:

$$Si \;\; t = 1 \;\; y \;\; a = 0, \; entonces \;\; {}_1w^{nuevo} = {}_1w^{anteior} + p \quad (2.1.43)$$

$$Si \;\; t = 0 \;\; y \;\; a = 1, \; entonces \;\; {}_1w^{nuevo} = {}_1w^{anterior} - p \quad (2.1.44)$$

$$Si \;\; t = a, \; entonces \;\; {}_1w^{nuevo} = {}_1w^{anterior} \quad (2.1.45)$$

Las tres condiciones anteriores pueden ser escritas en forma compacta (2.1.46) y generalizarse para la utilización de las funciones de transferencia *hardlim* o *hardlims*, generalización que es posible introduciendo el error en las reglas de aprendizaje del Perceptrón:

$$e = t - a \quad (2.1.46)$$

Por lo tanto, tenemos (2.1.47; 2.1.48; 2.1.49):

$$Si \; e = 1, \; entonces \;\; {}_1w^{nuevo} = {}_1w^{viejo} + p \quad (2.1.47)$$

$$Si \;\; e = -1, \; entonces \;\; {}_1w^{nuevo} = {}_1w^{anterior} - p \quad (2.1.48)$$

$$Si \;\; e = 0, \; entonces \;\; {}_1w^{nuevo} = {}_1w^{anterior} \quad (2.1.49)$$

En una sola expresión la ley puede resumirse (2.1.50) así:

$${}_1w^{nuevo} = {}_1w^{anterior} + ep = {}_1w^{anterior} + (t - a)p \quad (2.1.50)$$

Y extendiendo la ley a las ganancias (2.1.51)

$$b^{nueva} = b^{anterior} + e \quad (2.1.51)$$

Perceptrón multicapa:

El esquema general de un Perceptrón multicapa puede encontrarse generalizando la figura 2.1.52 a una red con múltiples entradas y que incluya una entrada adicional representada por la ganancia *b*, en donde se notan las conexiones entre sus nodos de entrada y las neuronas de salida.

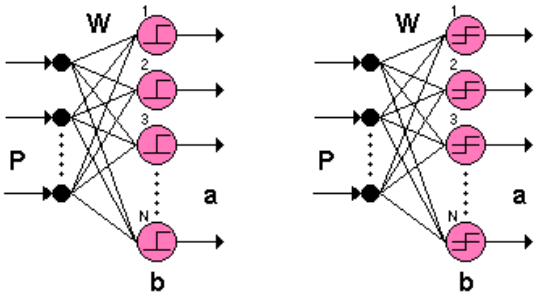


Figura 2.1.52 Conexiones del Perceptrón

Un Perceptrón multicapa es una red con alimentación hacia delante con capas ocultas, por lo cual establece regiones más complejas de decisión a diferencia de lo hace el Perceptrón de un solo nivel.

Un esquema simplificado del modelo del Perceptrón de la figura 2.1.52 se observa en la figura 2.1.53.

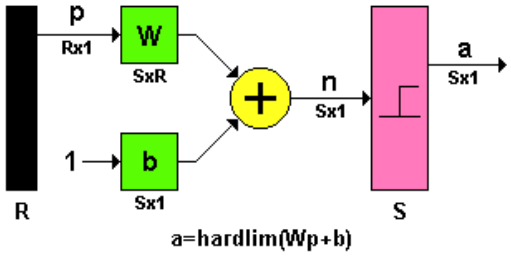


Figura 2.1.53 Notación compacta para la red tipo Perceptrón

La salida de la red está dada por (2.1.54):

$$a = hardlim\left(\boldsymbol{W} * \boldsymbol{p} + b\right) \quad (2.1.54)$$

Donde

***W*:** Matriz de pesos asignada a cada una de las entradas de la red de dimensiones SxR, con S igual al número de neuronas, y R la dimensión del vector de entrada

p: Vector de entradas a la red de dimensiones Rx1

***b*:** Vector de ganancias de la red de dimensiones Sx1

Las capacidades del Perceptrón multicapa con dos y tres capas y con una única neurona en la capa de salida se muestran en la figura 2.1.55 (de Hilera J y Martínez V). En la segunda columna se muestra el tipo de región de decisión que se puede formar con cada una de las configuraciones, en la siguiente se indica el tipo de región que se formaría para el problema de la XOR, en las dos últimas columnas se muestran las regiones formadas para resolver el problema de clases mezcladas y las formas más generales para cada uno de los casos. (Vemuri, 1990)

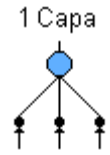
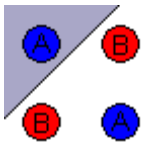
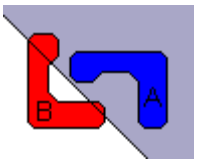
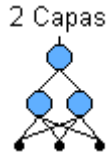
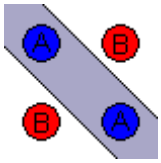
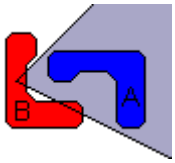
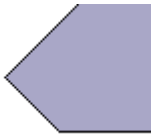
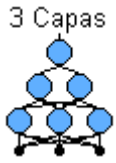
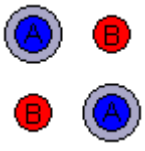
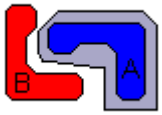
Estructura	**Regiones de Decisión**	**Problema de la XOR**	**Clases con Regiones Mezcladas**	**Formas de Regiones más Generales**
	Medio Plano Limitado por un Hiperplano			
	Regiones Cerradas o Convexas			
	Complejidad Arbitraria Limitada por el Número de Neuronas			

Figura 2.1.55 Distintas formas de las regiones generadas por un Perceptrón multicapa

El Perceptrón básico reconoce dos regiones de solución separadas por una línea; una red neuronal de perceptrones de dos capas forma regiones convexas arbitrarias como resultado de la intersección de regiones formadas por cada perceptron, activándose su salida para los patrones de un lado del hiperplano, si el valor de los pesos de las conexiones entre las neuronas de la segunda capa y una neurona del nivel de salida son todos igual a 1, y la función de salida es de tipo *hardlim*, la salida de la red se activará sólo si las salidas de todos los nodos de la segunda capa están activos, esto equivale a ejecutar la función lógica AND en el nodo de salida, resultando una región de decisión intersección de todos los semiplanos formados en el nivel anterior. La región de decisión resultante de la intersección será una región convexa con un número de lados a lo sumo igual al número de neuronas de la segunda capa.

A partir de este análisis surge el interrogante respecto a los criterios de selección para las neuronas de las capas ocultas de una red multicapa, este número en general debe ser lo suficientemente grande como para que se forme una región compleja que pueda resolver el problema, sin embargo, no debe ser muy grande pues la estimación de los pesos puede ser no confiable para el conjunto de los patrones de entrada disponibles. Hasta el momento no hay un criterio establecido para determinar la configuración de la red y esto depende más bien de la experiencia del diseñador. (Vemuri, 1990)

La regla de aprendizaje del Perceptrón para una red multicapa es una generalización de las ecuaciones (2.1.56) y (2.1.57)

$$_1\boldsymbol{W}^{nuevo} = {}_1\boldsymbol{W}^{anterior} + \boldsymbol{e}\boldsymbol{p}^T \quad (2.1.56)$$

$$\boldsymbol{b}^{nueva} = \boldsymbol{b}^{anterior} + \boldsymbol{e} \quad (2.1.57)$$

Backpropagation

La regla de aprendizaje del Perceptrón de Rosenblatt y el algoritmo LMS de Widrow y Hoff fueron diseñados para entrenar redes de una sola capa. Como se discutió anteriormente, estas redes tienen la desventaja que solo pueden resolver problemas linealmente separables, fue esto lo que llevo al surgimiento de las redes multicapa para sobrepasar esta dificultad en las redes hasta entonces conocidas.

El primer algoritmo de entrenamiento para redes multicapa fue desarrollado por Paul Werbos en 1974, este se desarrolló en un contexto general, para cualquier tipo de redes, siendo las redes neuronales una aplicación especial, razón por la cual el algoritmo no fue aceptado dentro de la comunidad de desarrolladores de redes neuronales. Fue solo hasta mediados de los años 80 cuando el algoritmo Backpropagation o algoritmo de propagación inversa fue redescubierto al mismo tiempo por varios investigadores, David Rumelhart, Geoffrey Hinton y Ronal Williams, David Parker y Yann Le Cun. El algoritmo se popularizó cuando fue incluido en el libro "ParallelDistributedProcessingGroup" por los sicólogos David Rumelhart y James McClelland. La publicación de este libro trajo consigo un auge en las investigaciones con redes neuronales, siendo la Backpropagation una de las redes más ampliamente empleadas, aun en nuestros días.

La mayoría de los sistemas actuales de cómputo se han diseñado para llevar a cabo funciones matemáticas y lógicas a una velocidad que resulta asombrosamente alta para el ser humano. Sin embargo, la destreza matemática no es lo que se necesita para solucionar problemas de reconocimiento de patrones en entornos ruidosos, característica que incluso dentro de un espacio de entrada relativamente pequeño, puede llegar a consumir mucho tiempo. El problema es la naturaleza secuencial del

propio computador; el ciclo tomar – ejecutar de la naturaleza Von Neumann solo permite que la máquina realice una operación a la vez. En la mayoría de los casos, el tiempo que necesita la máquina para llevar a cabo cada instrucción es tan breve (típicamente una millonésima de segundo) que el tiempo necesario para un programa, así sea muy grande, es insignificante para los usuarios. Sin embargo, para aquellas aplicaciones que deban explorar un gran espacio de entrada o que intentan correlacionar todas las permutaciones posibles de un conjunto de patrones muy complejo, el tiempo de computación necesario se hace bastante grande.

Lo que se necesita es un nuevo sistema de procesamiento que sea capaz de examinar todos los patrones en paralelo. Idealmente ese sistema no tendría que ser programado explícitamente, lo que haría es adaptarse a sí mismo para aprender la relación entre un conjunto de patrones dado como ejemplo y ser capaz de aplicar la misma relación a nuevos patrones de entrada. Este sistema debe estar en capacidad de concentrarse en las características de una entrada arbitraria que se asemeje a otros patrones vistos previamente, sin que ninguna señal de ruido lo afecte. Este sistema fue el gran aporte de la red de propagación inversa, Backpropagation.

La Backpropagation es una red neuronal de aprendizaje supérvisado con dos fases de adaptación: la primera fase inicia cuando se aplica el patrón en la entrada y este se propaga a través de todas las capasa hasta generar una salida y la segunda fase cuando se compara la salida real con la salida deseada y se calcula el error de cada salida.

El error de la salida se propaga hacia atrás desde de salida a las capas ocultas. Sin embargo, las neuronas de la capa oculta reciben una porción del error de acuerdo al peso con el que aportaron a la salida, esta acción se repite permitiendo actualizar los pesos de las neuronas hasta lograr el valor de la salida deseada que clasifiquen

correctamente todos los patrones de entrenamiento. De esta manera las capas ocultas logran reconocer características particulares de los patrones ingresados. Esto permite que las neuronas de las capas ocultas puedan reconocer características de los patrones arbitrarios que ingresen a la red activando la salida correspondiente.

Y a la inversa, las unidades de las capas ocultas tienen una tendencia a inhibir su salida si el patrón de entrada no contiene la característica para reconocer, para la cual han sido entrenadas.

Varias investigaciones han demostrado que, durante el proceso de entrenamiento, la red Backpropagation tiende a desarrollar relaciones internas entre neuronas con el fin de organizar los datos de entrenamiento en clases. Esta tendencia se puede extrapolar, para llegar a la hipótesis consistente en que todas las unidades de la capa oculta de una Backpropagation son asociadas de alguna manera a características específicas del patrón de entrada como consecuencia del entrenamiento. Lo que sea o no exactamente la asociación puede no resultar evidente para el observador humano, lo importante es que la red ha encontrado una representación interna que le permite generar las salidas deseadas cuando se le dan las entradas, en el proceso de entrenamiento. Esta misma representación interna se puede aplicar a entradas que la red no haya visto antes, y la red clasificará estas entradas según las características que compartan con los ejemplos de entrenamiento. (Vemuri, 1990)

Estructura de la Red: La estructura típica de una red multicapa se observa en la figura 2.1.58

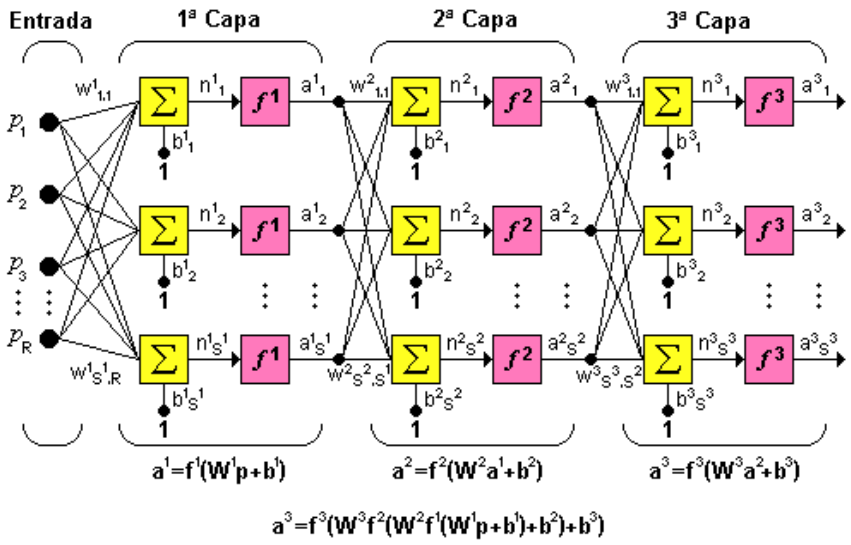


Figura 2.1.58 Red de tres capas

Puede notarse que esta red de tres capas equivale a tener tres redes tipo Perceptrón en cascada; la salida de la primera red, es la entrada a la segunda y la salida de la segunda red es la entrada a la tercera. Cada capa puede tener diferente número de neuronas, e incluso distinta función de transferencia.

En la figura 2.1.58 W^1 representa la matriz de pesos para la primera capa, W^2 los pesos de la segunda y así similarmente para todas las capas que incluya una red. Para identificar la estructura de una red multicapa, se empleará una notación abreviada, donde el número de entradas va seguido del número de neuronas en cada capa:

$$R : S^1 : S^2 : S^3 \quad (2.1.59)$$

Donde S representa el número de neuronas y el exponente representa la capa a la cual la neurona corresponde. (Vemuri, 1990)

La notación de la figura 2.1.59 es bastante clara cuando se desea conocer la estructura detallada de la red, e identificar cada una de las conexiones, pero cuando la red es muy grande, el proceso de conexión se torna muy complejo y es bastante útil utilizar el esquema de la figura 2.1.60

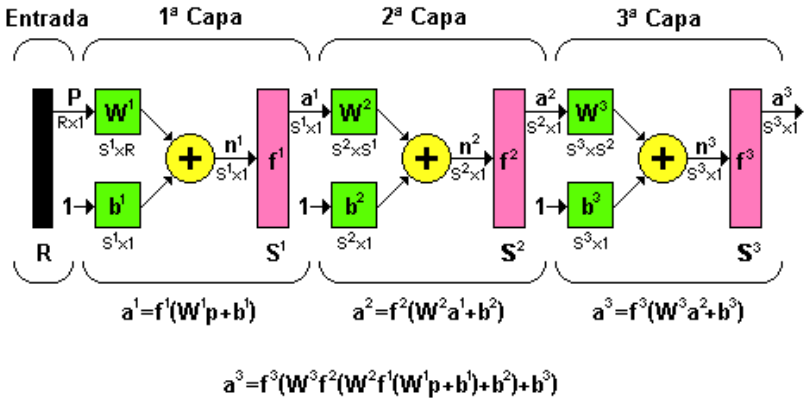


Figura 2.1.60 Notación compacta de una red de tres capas

Regla de Aprendizaje: El algoritmo Backpropagation para redes multicapa es una generalización del algoritmo LMS, ambos algoritmos realizan su labor de actualización de pesos y ganancias con base en el error medio cuadrático. La red Backpropagation trabaja bajo aprendizaje supervisado y por tanto necesita un set de entrenamiento que le describa cada salida y su valor de salida esperado. (Vemuri, 1990)

2.1.2 ANÁLISIS SINTÁCTICO

Definición del Análisis Sintáctico. -

El análisis Sintáctico es, en el campo de la Lingüística, el análisis de las funciones sintácticas o relaciones de concordancia y jerarquía que guardan las palabras agrupándose entre sí en sintagmas u oraciones. Como no está muchas veces claro el límite entre la sintaxis y la morfología a estos respectos, especialmente según el tipo de lengua de que se trate, también se suele denominar **análisis morfosintáctico**, aunque esta denominación se suele reservar para un análisis más profundo y detenido. (Baliri, 2014)

Definición del Analizador Sintáctico en los compiladores. - Un **analizador sintáctico** (o *parser*) es un programa informático que analiza una cadena de símbolos

de acuerdo a las reglas de una gramática formal. El término proviene del Latín pars, que significa parte (del discurso). Usualmente hace parte de un compilador, en cuyo caso, transforma una entrada en un árbol sintáctico de derivación.

El análisis sintáctico convierte el texto de entrada en otras estructuras (comúnmente árboles), que son más útiles para el posterior análisis y capturan la jerarquía implícita de la entrada. Un analizador léxico crea tokens de una secuencia de caracteres de entrada y son estos tokens los que son procesados por el analizador sintáctico para construir la estructura de datos, por ejemplo, un árbol de análisis o arboles de sintaxis abstracta. (Baliri, 2014)

El análisis sintáctico también es un estado inicial del análisis de frases de lenguaje natural. Es usado para generar diagramas de lenguajes que usan flexión gramatical, como los idiomas romances o el latín. Los lenguajes habitualmente reconocidos por los analizadores sintácticos son los lenguajes libres de contexto.

2.1.3 PROCESAMIENTO DEL LENGUAJE NATURAL(PLN)

El PLN se concibe como el reconocimiento y utilización de la información expresada en lenguaje humano a través del uso de sistemas informáticos.

En su estudio intervienen diferentes disciplinas tales como lingüística, ingeniería informática, filosofía, matemáticas y psicología. Debido a las diferentes áreas del conocimiento que participan, la aproximación al lenguaje en esta perspectiva es también estudiada desde la llamada ciencia cognitiva. (Cortez Vasquez, 2009)

Tanto desde un enfoque computacional como lingüístico se utilizan técnicas de inteligencia artificial:

- modelos de representación del conocimiento y de razonamiento,
- lenguajes de programación declarativos,

- algoritmos de búsqueda, y
- estructuras de datos.

Se investiga cómo el lenguaje puede ser utilizado para cumplir diferentes tareas y la manera de modelar el conocimiento.

El esquema general de la mayoría de los sistemas y métodos que involucran el procesamiento de lenguaje es el siguiente:

Primero, el texto no se procesa directamente sino se transforma en una representación formal que preserva sus características relevantes para la tarea o el método específico (por ejemplo, un conjunto de cadenas de letras, una tabla de base de datos, un conjunto de predicados lógicos, etc.).

Luego, el programa principal manipula esta representación, transformándola según la tarea, buscando en ella las subestructuras necesarias, etc.

Finalmente, si es necesario, los cambios hechos a la representación formal (o la respuesta generada en esta forma) se transforman en el lenguaje natural.

2.1.4 LA COMUNICACIÓN

El término lengua natural designa una variedad de lenguajes humanos con fines comunicativos que tiene una sintaxis y que obedece supuestamente a los principios de economía y optimidad. El lenguaje natural es la lengua o idioma hablado o escrito por humanos para propósitos generales de comunicación. Son aquellas lenguas que nacen generándose espontáneamente en un grupo de hablantes por la necesidad de comunicarse, a diferencia de otras lenguas, como puedan ser una lengua construida, los lenguajes de programación o los lenguajes formales usados en el estudio de la lógica formal, especialmente la lógica matemática. Y se realiza el estudio porque existe la necesidad de mejorar la comunicación hombre máquina.

2.1.4.1 Comunicación y Comunicación Humana

Según el *American College Dictionary* con de bastante amplitud. Comunicar es "formular o intercambiar pensamientos, opiniones o información de palabra, por escrito o a través de signos". Para cualquier definición será necesario puntualizar:

- Que la comunicación no es una faceta incidental de la vida, sino una función continua y esencial.
- Que, por tanto, no siempre es consciente, ni racional
- Que, biológicamente hablando, el estudio del origen y evolución de la comunicación no se restringe a la capacidad de producir y entender palabras *significativas*. Hay comunicación animal y comunicación no verbal.

Newton, en cambio, elevó el principio de acción y reacción a la categoría de axioma fundamental de la mecánica: "las acciones mutuas de dos cuerpos son siempre iguales y dirigidas en sentidos contrarios". El esquema acción/reacción, estímulo/respuesta, dominante durante dos siglos, redujo el problema de la comunicación a su dimensión física y mecanicista más elemental. Comunicación venía a ser todo intercambio o interacción entre dos cuerpos. Cuando los cuerpos son organismos vivos, el proceso de interacción se hace más complejo, pero obedece a la misma categoría trascendental, como vio Kant, al subsumir el principio de acción y reacción bajo la categoría relacional de *comunidad*. Estas contribuciones clásicas ponen las bases del tratamiento de la comunicación que actualmente llevan a cabo con gran precisión técnica la teoría de la información de Shannon y la Cibernética de Norbert Wiener:

El propio Weaver ensalza el valor general de la teoría matemática de la comunicación en los siguientes términos:

Es una teoría tan general que no se necesita fijar qué clase de símbolos se están considerando -si se trata de letras escritas o palabras, notas musicales, palabras habladas, música sinfónica, o cuadros.

En el curso de la comunicación, el mensaje puede ir cambiando de forma, y tales cambios deben ser ejecutados por los distintos transmisores. En el lenguaje oral, se considera al cerebro (o a la mente) como la fuente de información emisora, el transmisor es el mecanismo de la voz que produce variaciones de presión sonora (señal) que se transmiten por el aire; pero si hablo por teléfono se requiere un nuevo transmisor que convierta la presión sonora de la voz en una corriente eléctrica variable.

El receptor puede considerarse como el mecanismo inverso del transmisor, pues transforma o *decodifica* la señal transmitida, convirtiéndola en un mensaje inteligible para el *destinatario*. Cuando la comunicación se produce cara a cara interviene siempre un mecanismo de retroalimentación o de *feed-back*: el emisor regula continuamente su mensaje según las señales de comprensión que manifieste el destinatario. (Cortes, 1999)

El esquema de Shannon y Weaver goza de tal generalidad que puede aplicarse a cualquier tipo de comunicación. Desde el punto de vista de la teoría matemática de la comunicación sólo interesan, sin embargo, tres niveles relevantes según el tipo de problemas técnicos que planteen. Warren Weaver los denomina respectivamente problema técnico de *precisión*, problema semántico de *significado*, problema de influencia o de *efectividad*.

Estos tres niveles pueden coordinarse fácilmente con las dimensiones del lenguaje distinguidas por Morris: sintaxis, semántica y pragmática.

Desde el punto de vista de la naturaleza de los sistemas de comunicación existen, por lo menos, cuatro planos diferentes de análisis, que han dado lugar a diferentes áreas de investigación:

Plano intraorgánico: procesos que tienen lugar dentro de un organismo, como la recogida y elaboración de información biológica, psicológica, etc.

Plano interorgánico: procesos lingüísticos o psicológicos entre diferentes organismos.

Plano organizacional: procesos de comunicación institucionalizados, sistema de datos del mundo circundante de carácter más o menos interpersonal, etc.

Plano tecnológico: equipo, aparatos y programas establecidos para generar, almacenar, elaborar, transmitir, distribuir, etc., datos: la teoría de la información de Shannon y Weaver se mueve preferentemente en este plano, que traspasa e interfiere con los otros tres planos. (Cortes, 1999)

2.1.4.2 Funciones básicas de la comunicación

Distinguimos dos funciones en la comunicación de todo organismo:

Función adaptativa al medio circundante de objetos, y

Función modificadora de la conducta de otros organismos.

En el plano interorgánico o intercomunicacional cada una de estas funciones se desdobla en otras dos, a saber:

Función informativa: los sistemas vivos sólo tienen cierta viabilidad en sus entornos en la medida en que disponen de medios adecuados para adquirir y procesar información sobre sí mismos y sobre aquellos (a).

Función integrativa: acumulativa o autoorganizativa de los mensajes de los entornos. Se necesita para mantener el equilibrio y la estabilidad. Los mecanismos de *feed-back*juegan aquí un papel básico (a).

Funciones de mando e instrucción: características de las relaciones jerárquicas entre superior-subordinado y más patentes en las organizaciones formales (b).

Funciones de influencia y persuasión: características de las relaciones interpersonales a cualquier nivel, pero también de la política, la publicidad o la oratoria (b).

En todas estas funciones comunicativas predomina la dimensión pragmática. Las señales utilizadas en la mayor parte de las situaciones de comunicación animal cumplen con todo rigor estas funciones generales. Por ejemplo, cuando un individuo de una manada de gacelas percibe un peligro, huye, no sin antes emitir una señal que pone al resto de los animales sobre aviso y huyen en la misma dirección. (Cortes, 1999)

2.1.4.3 Características específicas de la comunicación humana.

Idealmente, la comunicación se perfecciona cuando se consigue una identidad de código entre emisor y destinatario; pero tal identidad sólo puede garantizarse plenamente cuando las señales que conforman el código son *inequívocas*, es decir, se corresponden unívocamente con los elementos cognoscitivos codificables y, sobre todo, cuando el conjunto de señales es *completo*. Inequivocidad y completud son, sin embargo, condiciones difícilmente alcanzables por los lenguajes naturales, siempre redundantes, ambiguos y abiertos a expansiones significativas insospechadas. De ahí que los códigos más perfectos desde este punto de vista sean los formalizados. Pero Gödel ha demostrado que ningún sistema formal lo suficientemente potente como para representar la aritmética elemental es completo. Dos cuestiones conectadas afloran a propósito de la posibilidad de construir una lengua universal en relación con la comunicación efectiva:

Los rasgos atribuidos a la buena comunicación colocan a los lenguajes naturales humanos en algún punto intermedio entre los lenguajes animales y los lenguajes de máquina. (Cortes, 1999)

Tomando como referencia nuestro sistema verbal, podemos definir los límites de la comunicación animal en términos de las propiedades que raramente o nunca se presentan. Así puede decirse que mientras los sistemas de comunicación animal son innatos, el lenguaje humano es adquirido. Esta primera diferencia no excluye, sin embargo, ni la posibilidad de aprendizaje de nuevos códigos ni la ampliación de señales por parte de los mamíferos superiores, con lo que la diferencia parece más de grado que de especie. Además, no puede olvidarse que los hombres están dotados de *competencias innatas*.

Una segunda diferencia, ligada con la anterior, reside en el hecho de que los lenguajes espontáneamente utilizados por los animales suelen poseer carácter mímico, mientras el habla humana posee siempre carácter articulado. Las señales integrantes de código mímico son ciertamente plurales, pero se reiteran de una manera fija y estereotipada por parte de todos los especímenes con posibilidades de pervivencia y perpetuación.

Por último, suele destacarse la recursividad heurística asociada al lenguaje humano, frente a la mera recursividad mecánica que las conductas animales presentan. Cualquier persona puede formar una frase que nadie haya empleado antes; más aún, tal creatividad aparece desde muy temprano como una característica diferenciadora de los especímenes humanos. (Chomsky, 1965)

El hombre, a su vez, en las sociedades complejas actuales, ha incrementado su capacidad de comunicación gracias a los portentosos avances tecnológicos de la ingeniería electrónica. Eso ha posibilitado la construcción de sistemas de comunicación

autorregulados, que interactúan y se comunican sin intervención humana, salvo en las fases iniciales y terminales del proceso. Aun siendo productos tecnológicos de la actividad humana, estos sofisticados sistemas de comunicación maquinal han alcanzado así un determinado grado de autonomía, que las diferencias de la comunicación específicamente humana. (Cortes, 1999)

Frente a ello, el lenguaje humano no es perfecto, sino infecto. *Los objetos del mundo se construyen a través de la actividad práctica humana in fieri*, siempre abierta, y nunca cancelada definitivamente. De ahí su riqueza y su superioridad, que se manifiesta en el hecho de que hayan sido los hombres quienes han construido los lenguajes artificiales de las máquinas y no al revés.

2.1.4.4 Evolución histórica del concepto de "comunicación"

El problema filosófico de la comunicación salta, como tal, a la escena filosófica con la llegada del existencialismo y la filosofía dialógica, en virtud del enfrentamiento crítico de estos movimientos con la tesis moderna, tanto racionalista como empirista, del aislamiento o separación, consustancial al hombre, respecto del ser (el problema del solipsismo), lo que vuelve problemático todo contacto entre mi yo y el yo de los otros. Sören Kierkegaard pone el énfasis en la exclusividad personal, no hay una apuesta posible por una comunicación eficiente. Kierkegaard parece que quiere un acceso directo a la divinidad. La grandeza humana (sobrehumana) no la encuentra en el estadio estético, ni en el ético, sino en el religioso. Para Gabriel Marcel, el mundo auténtico de vida humana es la comunicación, es la posibilidad suprema. (Kierkegaard, 2006)

La comunicación entre personas en Max Scheler puede darse como comunidad psíquica y vivencial en que se percibe al otro; o como relación entre sujetos que se

observan y razonan por analogía; y como vinculación interpersonal a la que corresponde la comprensión mutua. (Cortes, 1999)

En Martín Buber lo más destacado es la importancia suprema, constitutiva, que da al diálogo comunicativo entre interlocutores vueltos mutuamente el uno hacia el otro, expresándose sin reservas y libres de toda voluntad de aparentar o parecer. Para Ferdinand Ebner y D. von Hilebrand la comunicación es un salir de la soledad del yo hacia el tú por el habla, por el mero contacto, que permite percibir existencia mutas; por la unión de quienes buscan una convivencia y un conocimiento reales; y por la unificación propia del verdadero amor en que desaparece la resistencia mecánica del "yo-tú" para entrar en el nosotros.

McLuhan analizó la comunicación en función de los medios de comunicación; para él lo fundamental en la comunicación humana no es el significado ni la información misma, sino el medio de comunicación, o la estructura propia del medio, que impone las condiciones en que se lleva a cabo la información. Él ha hecho célebre la frase *the médium is the message*, que resume su tesis y que completa afirmando que *the médium is the masaje* (masaje, "manipulación", realizar con las manos). (Cortes, 1999)

2.1.5 EL LENGUAJE

El lenguaje tiene tres grandes funciones: expresión, cognición y conación. En la expresión lo fundamental es la traducción de emociones y necesidades; en la cognición, la aprehensión de la realidad; y en la conación la acción sobre otro. Pero las tres están muy relacionadas.

Para Vygotski el lenguaje surge como comunicación con los demás y sólo después, cuando se ha dominado su control, puede servir para hablar con uno mismo. Según Piaget, el pensamiento es anterior al lenguaje, por lo que primero pensamos, hablamos

con nosotros, y posteriormente lo hacemos con los demás. En opinión de Vygotski, el lenguaje se desarrolla con fines comunicativos antes que el aspecto intelectual. (Cortes, 1999)

2.1.5.1 La adquisición del lenguaje

Conductismo e innatismo. - Skinner defiende que el lenguaje se adquiere mediante refuerzos y condicionamiento. Para Chomsky, el lenguaje es innato, puesto que se despliega paulatinamente en el niño hasta que se fija. Chomsky considera que tenemos una estructura gramatical, de carácter mental, que condiciona el desarrollo del lenguaje. (Chomsky, 1965)

2.1.6 COMUNICACIÓN Y LENGUAJE

2.1.6.1 El uso del lenguaje en la comunicación humana: las funciones del lenguaje

Aunque es cierto que los animales transmiten señales informativas, éstas se limitan a situaciones y condiciones específicas y son poco flexibles. Se trata de un sistema cerrado, a diferencia de la comunicación humana, la cual *trasciende el esquema estímulo-respuesta y es un sistema abierto*, en el que se combinan libremente los símbolos. (Cortes, 1999)

La comunicación humana no tiene un carácter exclusivamente lingüístico. Entre los sistemas de comunicación no verbal destacan el lenguaje proxémico (relación de espacio físico y acción humana: proximidad corporal de interlocutores, códigos táctiles, visuales u olfaltivos, tono de voz, etc.), el cinésico (uso y movimiento del cuerpo en la comunicación), el gestual, el objetual y la presentación personal del yo o máscara (vestido, peinado, adorno, etc.). Se ha estimado que en una conversación entre dos interlocutores sólo el 35% del mensaje se realiza en palabras, mientras que el 65%

restante es comunicación no verbal. Ahora bien, el modo de comunicación más completo de que el hombre dispone es el lenguaje articulado.

La pragmática parte del supuesto de que la comunicación es la *función primaria* del lenguaje y, a partir de ahí, intenta especificar las*sub-funciones* principales que ésta cumple. Bühler establece la existencia de tres funciones principales del lenguaje: la representativa, la directiva y la expresiva. Jakobson añadió la función fáctica (comunicación mediante frases hechas y fórmulas ritualizadas), función poética (uso del lenguaje desde el punto de vista del estilo, sonoridad, etc.) y la función metalingüística (uso del lenguaje para hablar del lenguaje). (Cortes, 1999)

Las diversas funciones del lenguaje han sido usadas como criterio para clasificar los tipos de discurso: 1) *discurso declarativo*: es el constituido por las oraciones que, debido a lo que significan, pueden corresponder o no a lo que ocurre, a cómo son las cosas y pueden, en consecuencia, ser verdaderas o falsas; 2) *discurso prescriptivo*: está formado por aquellas oraciones que por lo que significan pueden ser cumplidas o incumplidas, según que se lleve o no a cabo lo que la oración dice; 3) *discurso expresivo*: consiste en oraciones que expresan, a causa de su significado, el estado psicológico del hablante o sus actitudes, y que por ello pueden ser sinceras o insinceras; 4) *discurso realizativo*: oraciones que, en virtud de lo que significan, enuncian el propio acto de habla que por medio de ellas se realiza.

2.1.6.2 Precondiciones de toda comunicación interpersonal

Los elementos que constituyen toda comunicación entre dos o más personas son:

Una relación estructurada por parámetros del tipo superior-subordinado, experto-novato, marido-mujer, etc.

Un conjunto de normas, comúnmente admitidas, psicológicas, sociológicas y antropológicas.

Un lenguaje entendido como la mutua aceptación de una moneda comunicativa o señal.

2.1.6.3 Tipos articulados de lenguaje humano

Entre los distintos tipos de lenguaje humano, los lingüistas privilegian la emisión vocal. Algunos estructuralistas justifican esta elección apelando a la idea platónica de *articulación*, como característica distintiva del lenguaje humano vocal primaria, secunda o derivadamente de la escritura alfabética.

Para el plano vocal puede introducirse la distinción lingüísticamente elaborada entre primera articulación de *monemas* (simples o lexemas y complejos o morfemas) y la segunda articulación de *fonemas*, finitos en número y dependientes del cálculo matemático de combinaciones, así como de limitaciones biológicas. En todo caso, cabe introducir el problema de la complejidad de este nivel hablado y de su dialéctica con la escritura.

2.1.6.4Teorías acerca de la naturaleza del lenguaje

Las posiciones son:

El lenguaje posee una naturaleza convencional. Defendida por Hermógenes el platónico, por Demócrito, los Sofistas, etc.; Aristóteles introduce una modulación moderada de la tesis, al introducir un tercer elemento, además del lenguaje y la realidad, a saber, el concepto formal o lógico. En el siglo XX, como continuación de la tradición nominalista y empirista reaparece la tesis en el segundo Carnap y en el segundo Wittgenstein. Entre los lingüistas estructuralistas, Martinet y Mounin subrayan la arbitrariedad claramente, y Sapir la introduce como rasgo de su definición.

El lenguaje tiene carácter natural. Tesis mantenida por Cratilo (Heráclito), que ha tenido numerosos seguidores. Tiene en común con la anterior la afirmación del carácter necesario de la relación entre el signo lingüístico y su objeto, pero se diferencian en el establecimiento y posibilidades de cambio de la asignación de nombres. Todos los naturalistas afirman que el lenguaje es apofántico, pero se diferencian entre sí al determinar el tipo de objetos que primariamente revelarían: I. La emotividad (teoría de la interjección de O. Jaspersen). II. Lo sensorialmente percibido (teoría onomatopéyica de Herder y mimética de Cassirer). III. Los pensamientos e ideas abstractas a través de metáforas (Teoría de Croce). IV. El lenguaje como imagen lógica del mundo o del ser (Estoicos, Fichte, primer Wittgenstein, Heidegger).

El lenguaje como instrumento, o sea, como producto de elecciones repetidas y repetibles. Es la tesis de Platón frente a las dos anteriores. Fue reproducida por Leibniz y defendida claramente por Humboldt, a través del cual la adoptará, en diversa medida, la lingüística contemporánea heredera de Saussure. Es en la tesis chomskiana de la *gramática generativa* donde esta tesis cobra su más profunda dimensión, al subrayar el aspecto creador del lenguaje a nivel de la utilización corriente. El sujeto hablante inventa y reinventa la lengua, instalándose en el plano de una combinatoria ilimitada, que recibiría de un código genético o gramática generativa universal.

El lenguaje como producto del azar, que propugna un estudio estadístico del lenguaje. Halla sus bases iniciales en las teorías de Shannon y Weaver.

2.1.6.5 Definición de lenguaje.

Una definición completa del lenguaje debe tener en cuenta no sólo sus aspectos gramaticales, sino también su dimensión física, biológica, psicológica y social:

Componentes físicos: el lenguaje como conjunto de señales acústicas o gráficas (signos) se halla sometido a restricciones ópticas, acústicas, mecánicas y termodinámicas.

Componentes biológicos: ciertas especializaciones de la anatomía y fisiología periféricas explican algunos rasgos universales de los lenguajes naturales. Pero el hecho de que individuos con serias anomalías periféricas (ciegos, sordos, anartríticos) puedan adquirir el dominio del lenguaje, indica que en el estado actual el factor determinante de la conducta lingüística es la función cerebral. Componentes psicológicos: el rasgo de creatividad o de la capacidad de producir mensajes nuevos y entenderlos (recursividad heurística), aun cuando de ellos no haya existido experiencia previa, tanto a nivel sintáctico (Chomsky), como semántico.

Componentes sociales: a nivel de adquisición de lenguaje, juegan el papel de mecanismos disparadores de la competencia lingüística.

2.1.7 EL PRINCIPIO DE LA PERTINENCIA Y LA INTERPRETACIÓN DE LAS LOCUCIONES

La información que codifica cualquier locución es compatible con un conjunto infinito de posibles pensamientos o "mensajes" que, a su vez, pueden entrar en interacción con un conjunto infinito de posibles supuestos, de los que el oyente podría derivar innumerables nuevos supuestos. Estas decisiones, que son de diversos tipos, afectan a los dos niveles básicos de la comunicación intencional y abierta: el componente explícito y el componente implícito

La *elección del contexto adecuado* es determinante para dar con la interpretación correcta.

2.1.8 PENSAMIENTO Y LENGUAJE

La relación entre las palabras y las cosas, esto es, el problema de la *referencia*, remite de inmediato a la mediación que suponen los procesos cognitivos del hombre, mediante los cuales organiza su experiencia sensorial. La teoría de la *abstracción* aristotélica juega aquí un papel fundamental y ha sido contrastada experimentalmente. La actividad de nombrar puede considerarse como la peculiaridad humana para hacer explícito un proceso que es bastante universal entre los animales superiores: la organización de los datos sensoriales. Esto supone una categorización de los estímulos o una etiquetación de la realidad, a la que siguen procesos de transformación y de diferenciación. Ello parece dar prioridad al pensamiento sobre el lenguaje. Pero, por otro lado, como decía Rousseau, "hay que enunciar proposiciones para tener ideas generales", apreciación aparentemente confirmada por el aprendizaje escolar.

2.1.9 LA IDEA DEL LENGUAJE A TRAVÉS DEL ESTUDIO DE SUS FUNCIONES

2.1.9.1 La aproximación lingüística de Jakobson

Para Jakobson el lenguaje debe ser estudiado en la variación de sus funciones, que él extrae del análisis de los factores constitutivos del mismo, a saber, el mensaje, el remitente, el destinatario, el establecimiento de contactos entre ellos, el contexto del mensaje y el código, que permite descifrarlo. De aquí surgen seis funciones:

Función expresiva

Función conativa (= apelativa)

Función denotativa, cognitiva o referencial

Función fática o de contacto,

Función poética o literaria,

Función metalingüística,

2.1.9.2La semiótica de Morris

Concibe el lenguaje como conjunto de signos cuya consideración global correspondería a la Semiótica. Pero hay tres términos en la relación tríadica de la semiosis: *signo, designatum, intérprete*. De esta relación tríadica pueden ser extraídas para las siguientes relaciones diádicas:

La relación semántica,

La dimensión pragmática,

El contexto sintáctico,

2.1.9.3 La aproximación de Karl Bühler

En su *Teoría del lenguaje* parte Bühler de la triple distinción psicológica entre *efecto, impulso* y *conocimiento* para distinguir las siguientes funciones, que varían, en cada caso particular, en proporciones diversas.

Las tres dimensiones del lenguaje, según Bühler, son:

Dimensión expresiva o emotiva:

Dimensión interpelativa, apelativa o deíctica:

Dimensión representativa, referencial o simbólica:

2.1.9.4 Los actos de habla

La propuesta del segundo Wittgenstein de que el significado de una expresión se determina por el uso que los hablantes hacen de ella se convierte en el punto de partida del desarrollo de la concepción del significado que se conoce como la *teoría de los actos de habla*. La estrategia de Austin (su formulador) es considerar el lenguaje como un instrumento *para hacer cosas*. Distingue tres actos de habla:

Acto ilocucionario o locutivo:

b) *acto fático*,

c) *acto rético*,

2.1.10 LENGUAJE NATURAL Y LENGUAJE ARTIFICIAL

El uso cotidiano y común del lenguaje posee unos significados y unas estructuras poco definidas en los que abundan las insuficiencias, las reducciones, las ambigüedades y las segundas intenciones. Por eso, desde el punto de vista semántico son frecuentes los términos equívocos y las polisemias, es decir, las palabras que sirven para designar objetos distintos, otros se refieren a cualidades imprecisas o espurias ("difícil", "ameno", "aceptable"…), los hay que designan objetos o realidades abstractos ("lujuria","templanza","justicia",..)o sentimientos y estados afectivos imposibles de precisar ("melancolía", "ansiedad", ..)

Los lenguajes artificiales o convencionales propiamente dichos, por su parte, se elaboran intencionalmente con miras a determinados fines o en virtud de una necesidad expresiva muy precisa. Desde este punto de vista, los lenguajes artificiales sólo resultan posibles gracias a la capacidad reflexiva de los lenguajes naturales, es decir, debido a la potencia que poseen los lenguajes naturales para examinar, analizar y precisar sus propios contenidos, estructuras y significados.

En general, estos lenguajes se caracterizan por su rigor y exactitud y, en este sentido, constituyen medios o instrumentos elaborados por los científicos, que permiten expresar de manera adecuada los objetos estudiados por sus ciencias.

Con respecto al lenguaje natural, el lenguaje formal presenta las siguientes ventajas:

Es posible dar a los símbolos lógicos un significado absolutamente preciso, cosa que no tienen generalmente los términos del lenguaje ordinario. Los términos del lenguaje

normal tienen muy a menudo los "bordes gastados", es decir, los límites de su connotación y de su denotación son muy imprecisos.

2.1.10.1Las categorías de un lenguaje formal

La diferencia fundamental entre un lenguaje natural y un lenguaje formal radica en su estructura. Un lenguaje formal (al que en lo sucesivo denominaremos cálculo) es un sistema de símbolos definidos con precisión, más unas reglas de formación de expresiones mediante la combinación de estos símbolos, y unas reglas de transformación de esas expresiones en otras expresiones del cálculo. Las expresiones bien formadas, esto es, las formadas de acuerdo con las reglas, se denominan fórmulas del cálculo. A un cálculo le atañen únicamente consideraciones sintácticas del tipo "qué es un signo del cálculo y qué no lo es", "qué símbolos son primitivos y cuáles derivados", etc. Los símbolos del cálculo en rigor carecen de significado, esto es no se refieren a nada, por lo que un cálculo en cuanto tal está semánticamente vacío. Por tanto, mientras el cálculo sea un puro cálculo, sus símbolos no son propiamente símbolos -pues no representan nada- sino meros grafismos.

2.1.10.2 Usos de la formalización

Históricamente la simbolización ha sido muy posterior a la formalización, pues esta última se consiguió en la lógica con Aristóteles, mientras que la primera hubo de esperar a la mitad del siglo XIX con Boole. Formalizar es reemplazar en el discurso los términos con significado categoremático (sustantivos, adjetivos, verbos y adverbios) por variables carentes de tal significado (por ejemplo, reemplazar "Todos los hombres son mortales" por "Todos los X son Y"). Simbolizar es reemplazar los términos del lenguaje natural que aún quedan tras la formalización (los denominados sincategoremáticos, como determinadas preposiciones -las conectivas- y adjetivos -los

cuantificadores-) por determinados signos. Evidentemente, con la sola formalización y simbolización de expresiones no tenemos aún un lenguaje formal, es menester dotar a ese lenguaje de la estructura calculística.

III MÉTODO

3.1 TIPO DE INVESTIGACION

La presente investigación según las variables que los representan es experimental y según el nivel de medición y el análisis de la información es de nivel IV porque es el aporte de un nuevo conocimiento destinado a procurar soluciones de problemas prácticos.

La presente investigación es de nivel Predictiva II por contener un diseño experimental el cual permitirá demostrar la hipótesis y luego de las formalizaciones de las estructuras sintácticas del Idioma castellano se logrará el procesamiento de dichas estructuras en un prototipo.

3.2 POBLACION Y MUESTRA

La población es un prototipo de software que se desarrolló para hacer las pruebas a las estructuras sintácticas del Idioma castellano que han sido formalizadas, en formas de las oraciones (Nominativo, Genitivo, Dativo, Acusativo, Vocativo, Ablativo) en formato escrito del idioma castellano.

La muestra es toda la población, es decir el prototipo de analizador sintáctico.

3.3 OPERACIONALIZACION DE VARIABLES

Variable independiente. - La variable independiente es: **MÉTODOS Y TÉCNICAS DE APLICACIÓN DE LAS REDES NEURONALES ARTIFICIALES PARA REALIZAR EL ANALISIS SINTACTICO del lenguaje natural del idioma castellano**

Variable dependiente. - La variable dependiente es: **EFICACIA EN EL RECONOCIMIENTO DE LAS ESTRUCTURAS GRAMATICALES DEL IDOMA CASTELLA –ANALISIS SINTACTICO.**

Indicadores de la hipótesis general

Indicador 1

Cantidad de respuestas del sistema para hacer el reconocimiento de las estructuras sintácticas.

Indicador 2

Precisión en el reconocimiento de la estructura sintáctica.

Indicador de la hipótesis secundaria

Indicador 1

Cantidad de respuestas del sistema para hacer el reconocimiento de las palabras del idioma castellano.

Indicador 2

Precisión en el reconocimiento de las palabras del idioma castellano.

3.4 INSTRUMENTOS

Para la recolección de datos teóricos científicos se utilizó la exploración conceptual de materiales bibliográficos y para la demostración de la hipótesis se obtuvieron los datos experimentales en el prototipo tecnológico desarrollado en para la presente investigación. Por otro lado, se realizó:

- La exploración conceptual para diseñar la estructura de la red neuronal artificial.
- La exploración conceptual para diseñar los algoritmos de aprendizaje y funcionamiento de la red neuronal artificial.

- La exploración conceptual y experimental para determinar las herramientas tecnológicas de programación para implementar la solución al problema planteado en la presente investigación.

  Para la contratación de la hipótesis se utilizó resultados de la implementación experimental del prototipo que permitió verificar el reconocimiento de las oraciones en los distintos casos formalizados para el analizador sintáctico

3.5 PROCEDIMIENTOS

Para el desarrollo de la tesis se realizó las siguientes fases:

- **Se estudiaron las distintas palabras del lenguaje natural (Castellano, determinando características particulares. -** En esta fase de la identificación de las distintas palabras del lenguaje castellano, se clasifico las palabras y se codifico para su uso.
- **Se creó una base de datos con las palabras del lenguaje natural (castellano). -** Se desarrolló una base de datos con las palabras del lenguaje natural (castellano) y se recopilo las palabras.
- **Se identificó las distintas estructuras sintácticas del lenguaje natural (castellano). -** En esta fase se identificó las distintas estructuras sintácticas del lenguaje castellano.
- **Se diseñó la red neuronal artificial para realizar el análisis sintáctico del lenguaje natural. –**Se determinó la estructura de la red neuronal artificial para realizar el análisis sintáctico del lenguaje natural (castellano).

- **Se diseñó de algoritmo de aprendizaje de la red neuronal artificial para realizar el análisis sintáctico del lenguaje natural (castellano). -**Se diseñó el algoritmo de aprendizaje para la estructura de red del punto anterior.
- **Se desarrolló y se hicieron las pruebas al software del algoritmo de aprendizaje de la red neuronal artificial para realizar el análisis sintáctico del lenguaje natural (castellano). -**Se realizó la programación del algoritmo de aprendizaje en un lenguaje de programación, y se realizó las pruebas correspondientes para determinar la convergencia del algoritmo.
- **Se diseñó el algoritmo de funcionamiento de la red neuronal artificial para realizar el análisis sintáctico del lenguaje natural (castellano). -**Se diseñó el algoritmo de funcionamiento de la red neuronal artificial.
- **Se desarrolló y se hicieron pruebas al software del algoritmo de aprendizaje de la red neuronal artificial para realizar el análisis sintáctico del lenguaje natural (castellano). -**Se programó en un lenguaje de programación el algoritmo de funcionamiento de la red neuronal artificial y se realizara las pruebas correspondientes.
- **Se hicieron pruebas para probar la eficacia del software que realizo el análisis sintáctico del lenguaje natural (castellano). -** Se realizó con el software una serie de pruebas con distintas estructuras y se calculó la eficacia del software.

  Estas fases se muestras a continuación.

3.5.1 Formalización de las oraciones escritas para el lenguaje natural

El lenguaje Natural es un lenguaje utilizado por las personas para comunicarse, el castellano corresponde a uno de los lenguajes naturales en contraposición con los lenguajes formales (aquellos usados en la programación lógica.)

Para lograr el procesamiento del lenguaje natural (castellano) será necesario primero lograr la formalización del lenguaje de tal manera que a partir de una frase escrita o hablaba se pueda efectuar acciones adecuadas a la información recibida.

Las palabras del idioma castellano presentan algunas características específicas que permite distinguirlas unas de otras por ejemplo la palabra colegio no es la misma que la palabra estudio ni tampoco que la palabra maravilloso. Por ello en tal sentido el castellano tiene nueve diferentes tipos de palabras: articulo, sustantivo, adjetivo, verbo, pronombre, adverbio, preposición, conjunción e interjección.

Para poder realizar un analizador sintáctico, usaremos las herramientas que se emplean en el estudio de los lenguajes formales y con ello lograremos formalizar el lenguaje natural Castellano para ello se consideró que lo primero que tenemos que reconocer son los tipos de estructuras del lenguaje, las cuales están conformadas por las distintas palabras del lenguaje a estructurar, en este caso el Castellano, por ellos de las nueve tipos de palabras que tiene el idioma castellano solo utilizaremos 5 tipos (Artículos, sustantivo, adjetivo, verbo, pronombre).

En una segunda etapa se analizó todos los formatos de oraciones llegando a la conclusión de que los tipos de oraciones que se iban a requerir solo eran en forma de ordenes en diferentes tiempos y numero como, por ejemplo:

- Abre el archivo tesis.

Como se puede ver es una oración que denota una orden, que se podría dar tanto a una persona como a una máquina, de ahí que es la forma de oración requerida para

que nuestro analizador sintáctico pudiera reconocer que es un formato valido. De la oración se puede apreciar que el formato aceptado seria ver figura 3.5.1

Abre →verbo regular tiempo presente tercera persona

El→ articulo determinado

Archivo→sustantivo común

Tesis→sustantivo Propio

De ahí se va creando una estructura como:

VRP-P3 AD SC SP

Figura 3.5.1 Estructura sintactica formalizada

En el anexo 1 se podra ver todas los formatos y oraciones usadas.

3.5.2 Analizador Sintactico para el Lenguaje Natural

El analizador sintactico propuesto en la presente tesis para realizar el reconocimiento de las estructuras sintacticas formalizadas se muestra en el diagrama de la figura 3.5.2 el cual consta de dos partes: (Borja Mario, Torres sally, Lescano sergio, 2009)

- Aprendizaje
- Funcionamiento

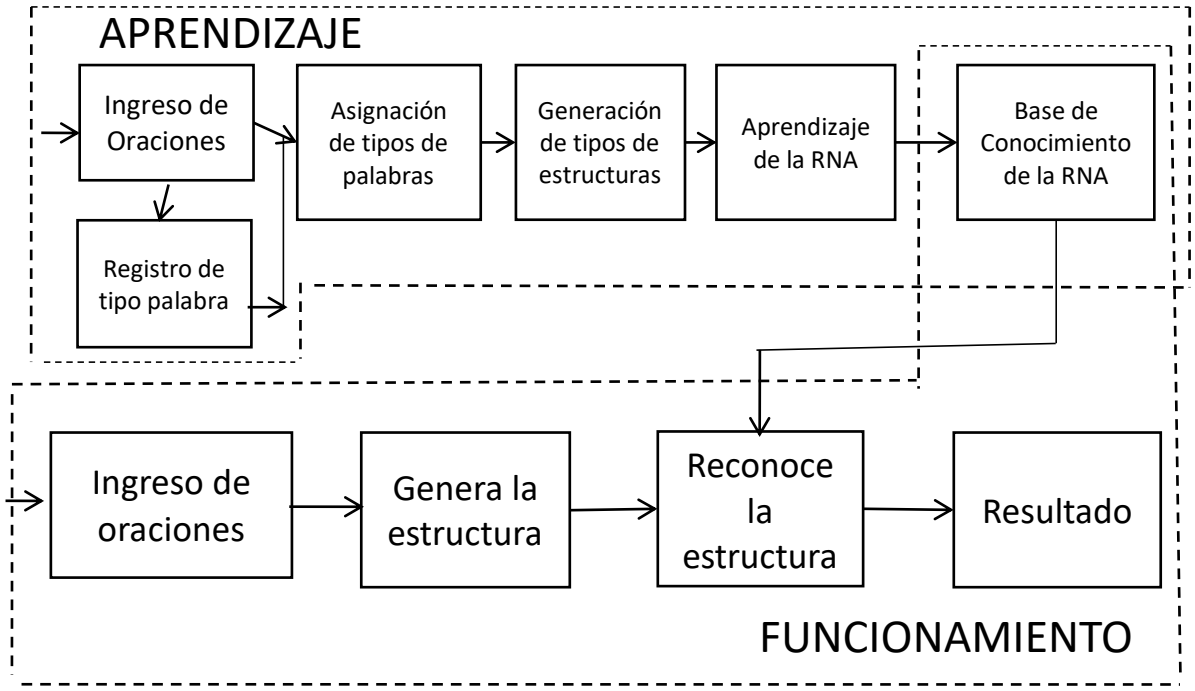


Figura 3.5.2 Diagrama del analizador sintactico para realizar el reconocimiento de las estructuras sintacticas formalizadas.

Los componentes de la parte del Aprendizaje del sistema son los siguientes:

Ingreso de Oraciones

Se trata del proceso de ingreso de oraciones a través de una interfaz creada en el prototipo de prueba, como se muestra en la figura 3.5.3, para luego ser procesadas por el sistema.

Figura 3.5.3 Interface para el ingreso de oraciones

Registro de Tipos de Palabras

Es un proceso mediante el cual se va registrando a cada palabra de las oraciones ingresadas su respectivo tipo. Tal como se muestra en la figura 3.5.4

Figura 3.5.4 Registro del tipo de palabra

Asignación de Tipos de Palabras

Mediante este proceso se va asignando el tipo de palabras (verbo en que forma y tiempo, sustantivo, articulo, etc) a las palabras de las oraciones como se muestra en

la figura 3.5.5, de tal manera que se vaya creando una base de datos a partir de donde luego se utilizara para convertir las oraciones a estructuras.

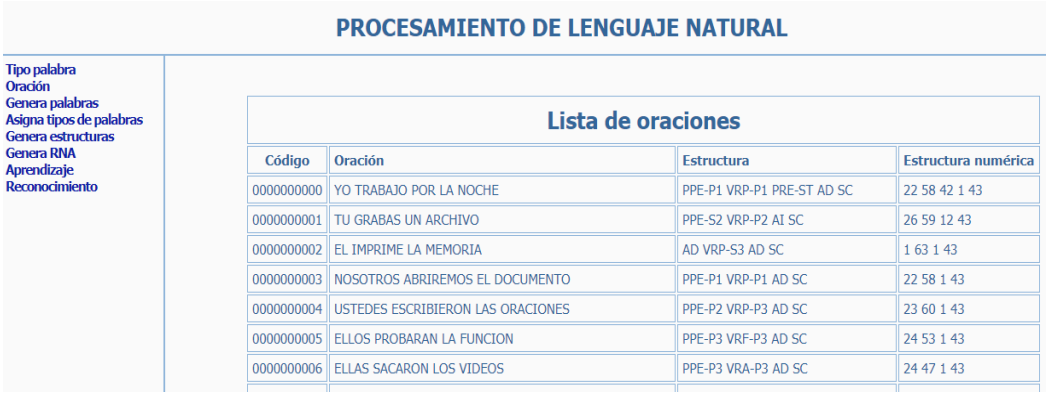
Lista de palabras

Código	Palabra	Tipo palabra
0000000000	YO	PPE-P1 : Pronombre Personal Plural 1ra Persona Cambiar tipo de palabra
0000000001	TRABAJO	VRP-P1 : Verbo Regular Presente Plural 1ra Persona Cambiar tipo de palabra
0000000002	POR	PRE-ST : Preposición Separable de Tiempo Cambiar tipo de palabra
0000000003	LA	AD : Articulo Determinado Cambiar tipo de palabra
0000000004	NOCHE	SC : Sustantivo Común Cambiar tipo de palabra
0000000005	TU	PPE-S2 : Pronombre Personal Singular 2da Persona Cambiar tipo de palabra

Figura 3.5.5 Asignación de tipos de palabras

Generación de las estructuras

Este es un proceso mediante el cual de las oraciones proporcionadas en lenguaje escrito el sistema les va asignando a qué tipo de palabra corresponde, de la teoría de compiladores se dice que se encuentra los tokens con su respectivo parámetros tipo y valor. Para nuestro caso será Palabra y tipo de palabra. Este proceso se puede ver en la figura 3.5.6.



PROCESAMIENTO DE LENGUAJE NATURAL

Tipo palabra
Oración
Genera palabras
Asigna tipos de palabras
Genera estructuras
Genera RNA
Aprendizaje
Reconocimiento

Lista de oraciones

Código	Oración	Estructura	Estructura numérica
0000000000	YO TRABAJO POR LA NOCHE	PPE-P1 VRP-P1 PRE-ST AD SC	22 58 42 1 43
0000000001	TU GRABAS UN ARCHIVO	PPE-S2 VRP-P2 AI SC	26 59 12 43
0000000002	EL IMPRIME LA MEMORIA	AD VRP-S3 AD SC	1 63 1 43
0000000003	NOSOTROS ABRIREMOS EL DOCUMENTO	PPE-P1 VRP-P1 AD SC	22 58 1 43
0000000004	USTEDES ESCRIBIERON LAS ORACIONES	PPE-P2 VRP-P3 AD SC	23 60 1 43
0000000005	ELLOS PROBARAN LA FUNCION	PPE-P3 VRF-P3 AD SC	24 53 1 43
0000000006	ELLAS SACARON LOS VIDEOS	PPE-P3 VRA-P3 AD SC	24 47 1 43

Figura 3.5.6 Generación de la estructura

Aprendizaje de la RNA

Es un proceso mediante el cual se ajustan los pesos de la red (en total 3000000 de pesos) utilizando las diversas muestras para que pueda realizar el reconocimiento de las estructuras sintácticas formalizadas. Se puede apreciar en la figura 3.5.7. y 3.5.8.

Figura 3.5.7 Generación del tipo de palabra

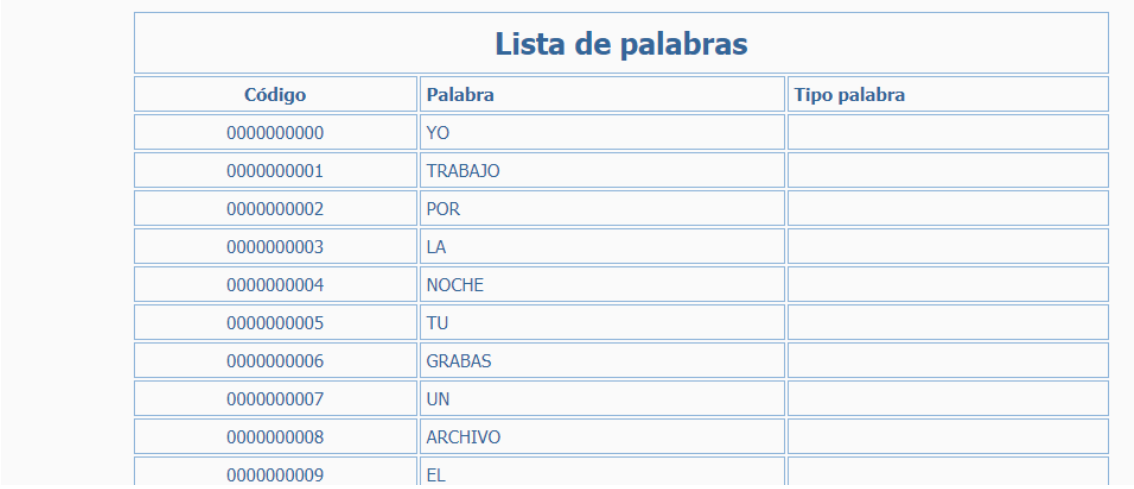


Figura 3.5.8 reconoce el tipo de palabra.

Base de conocimiento de la RNA

En la base de conocimentos se almacenan el conocimiento de la RNA que va obteniendose cuando se realiza el proceso de aprendizaje del conjunto de oraciones ingresadas en la interface en formato escrito.

Los componentes de la parte del **Funcionamiento** del sistema son los siguientes:

Ingreso de Oraciones

Se trata del proceso de ingreso de oraciones a través de una interfaz creada en el prototipo de prueba, como se muestra en la figura 3.5.3, para luego ser procesadas por el sistema. (cabe señalar que el formato del ingreso de las oraciones es igual al mencionado en el proceso uno de la parte de Aprendizaje).

Genera la estructura.

Mediante este proceso utilizando los tipos de palabras ingresadas en el proceso de aprendizaje se genera la estructura sintáctica correspondiente.

Reconocimiento de la Estructura

Este es un proceso mediante el cual la RNA utilizando el conocimiento obtenido durante el proceso de aprendizaje reconocerá la estructura sintáctica ingresada.

Resultado

Es un proceso mediante el cual se muestra el resultado obtenido en la pantalla al usuario.

3.5.3 Desarrollo de Prototipo de Analizador Sintáctico. –

Se desarrolló un software en PHP que consta de 7 módulos los cuales son tal como se muestra en el diagrama del analizador sintáctico más, un módulo que se encarga de la generación de la RNA.

Base de Datos. - la base de datos esta compuestas por las tablas**:** oración, palabra, tipo palabra, estructura-neurona, neurona, peso. A continuación, se muestra la base de datos en mysql.

```
/*
```

```
SQLyog - Free MySQL GUI v5.15 Host - 5.5.43-MariaDB : Database - ln
*********************************************************************
Server version : 5.5.43-MariaDB
*/
SET NAMES utf8;
SET SQL_MODE='';
create database if not exists `ln`;
USE `ln`;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO';
/*Table structure for table `estructura_neurona` */
DROP TABLE IF EXISTS `estructura_neurona`;
CREATE TABLE `estructura_neurona` (
  `id_estructura_neurona` int(11) NOT NULL,
  `dato_estructura_neurona` int(11) DEFAULT NULL,
  PRIMARY KEY (`id_estructura_neurona`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
/*Table structure for table `neurona` */
DROP TABLE IF EXISTS `neurona`;
CREATE TABLE `neurona` (
  `m` int(11) DEFAULT NULL,
  `capa` int(11) DEFAULT NULL,
  `n` float DEFAULT NULL,
  `a` float DEFAULT NULL,
  `k` float DEFAULT NULL,
  `u` float DEFAULT NULL,
  `id_neurona` int(10) unsigned NOT NULL,
```

```
  PRIMARY KEY (`id_neurona`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
/*Table structure for table `oracion` */
DROP TABLE IF EXISTS `oracion`;
CREATE TABLE `oracion` (
  `codigo_oracion` varchar(10) CHARACTER SET latin1 COLLATE latin1_spanish_ci NOT NULL,
  `oracion` varchar(250) CHARACTER SET latin1 COLLATE latin1_spanish_ci DEFAULT NULL,
  `estructura` varchar(250) CHARACTER SET latin1 COLLATE latin1_spanish_ci DEFAULT NULL,
  `estructura_numerica` varchar(250) CHARACTER SET latin1 COLLATE latin1_spanish_ci DEFAULT NULL,
  PRIMARY KEY (`codigo_oracion`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
/*Table structure for table `palabra` */
DROP TABLE IF EXISTS `palabra`;
CREATE TABLE `palabra` (
  `codigo_palabra` varchar(10) CHARACTER SET latin1 COLLATE latin1_spanish_ci NOT NULL,
  `palabra` varchar(45) CHARACTER SET latin1 COLLATE latin1_spanish_ci DEFAULT NULL,
  `codigo_tipo_palabra` varchar(10) CHARACTER SET latin1 COLLATE latin1_spanish_ci NOT NULL,
  PRIMARY KEY (`codigo_palabra`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
/*Table structure for table `peso` */
DROP TABLE IF EXISTS `peso`;
CREATE TABLE `peso` (
  `id_peso` int(11) NOT NULL,
```

```
  `id_neurona` int(11) NOT NULL,
  `valor_peso` float DEFAULT NULL,
  PRIMARY KEY (`id_peso`,`id_neurona`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
/*Table structure for table `tipo_palabra` */
DROP TABLE IF EXISTS `tipo_palabra`;
CREATE TABLE `tipo_palabra` (
  `codigo_tipo_palabra` varchar(10) CHARACTER SET latin1 COLLATE latin1_spanish_ci
DEFAULT NULL,
  `descripcion_tipo_palabra` varchar(45) CHARACTER SET latin1 COLLATE latin1_spanish_ci
DEFAULT NULL,
  `numero_tipo_palabra` int(11) NOT NULL AUTO_INCREMENT,
  PRIMARY KEY (`numero_tipo_palabra`)
) ENGINE=InnoDB AUTO_INCREMENT=64 DEFAULT CHARSET=latin1;
SET SQL_MODE=@OLD_SQL_MODE;
```

Módulo de ingreso de oraciones. -

La interface en donde se realiza este proceso se mostró en la figura 3.5.3 el código es el siguiente:

```
function FormOracion($db){
        $oracion = new formato();
        $html = $oracion->titulo("Registro de Oraciones","FOracion");
        $html = $html.$oracion->texto("Oración","oracion","Carga el word","90");
        $ruta1="index.php?cmd=grabaOracion";
        $ruta2="index.php?cmd=listaOracion";
        $graba="\"javascript:grabaOracion(FOracion,'".$ruta1."','".$ruta2."','contenido','listacont'
)\"";
        $html = $html.$oracion->boton($graba);
        echo $html;
}

function listaOracion($db){
        $html= "<table border=0 align=center>
```

```
            <tr>
            <td colspan=3 class=titulo>Oraciones</td>
            </tr>";
            try {
            $st = $db->prepare("select codigo_oracion,oracion from oracion");
           $st->execute();
           $html=$html."<br><br><tr>
           <td class=campo></td>
                    <td class=cabCentro>Código</td>
                    <td class=cabIzquierda>Oración</td>
             </tr>";

           while ($result = $st->fetch(PDO::FETCH_ASSOC)) {
                 $ruta = "index.php?cmd=actualizaOracion&codigo=".$result['codigo_oracion'];
                 $html=$html."<tr>
                          <td class=filaDerecha>
                          <a href=\"#\" onclick=\"javascript:cargaRuta('".$ruta."','contenido')\">
                          <img src=images/editar.svg alt=\"Actualizar datos\" height=30 width=30
border=0 title=\"Actualizar datos\">
                           </a>
                        </td>
           <td class=filaCentro>".$result['codigo_oracion']."</td>
                    <td class=filaIzquierda>".$result['oracion']."</td>
                    </tr>";
           }
           $html=$html."</table>";
}
catch (Exception $e) {
         Echo "Failed: " . $e->getMessage();
       }
        echo $html;
}
function grabaOracion($oracion,$db){
           $basic = new datosbasicos();
           $oracionM = strtoupper($oracion);
           $codigo = $basic->generaCodigo("oracion","codigo_oracion",$db);
           $oracion = new formato();
            try {
                $stmt = $db->prepare("INSERT INTO oracion (codigo_oracion,oracion)
                          VALUES ('$codigo','$oracionM')");
                        $db->beginTransaction();
                        $stmt->execute();
                        $db->commit();
           } catch (Exception $e) {
           $db->rollBack();
           echo "Failed: " . $e->getMessage();
           }
           $html = $oracion->tituloPresenta("Datos de la Oración","Foracion");
```

```
            $html = $html.$oracion->presenta("Codigo",$codigo);
            $html = $html.$oracion->presenta("Oracion",$oracion);
            $html = $html.$oracion->pie("Se grabo la oración");
             echo $html;
}

function actualizaOracion($codigo,$db){
 try {
     $st = $db->prepare("select codigo_oracion,oracion from oracion where
codigo_oracion='$codigo'");
     $st->execute();
     $result = $st->fetch(PDO::FETCH_ASSOC);
     $oracion = $result['oracion'];
 }
 catch (Exception $e) {
         Echo "Failed: " . $e->getMessage();
}
 $actuaOracion = new formato();
 $html = $actuaOracion->titulo("Actualiza la oracion","Foraciona");
 $html = $html.$actuaOracion->atexto("Oracion","oracion",$oracion,"60");
 $ruta1="index.php?cmd=grabaActualizaOracion&codigo=".$codigo;
 $ruta2="index.php?cmd=listaOracion";
 $graba="\"javascript:grabaOracion(Foraciona,'".$ruta1."','".$ruta2."','contenido','listacont')\"";
 $html = $html.$actuaOracion->boton($graba);
 echo $html;
}

 function grabaActualizaOracion($codigo,$oracion,$db){
            $orac = new formato();
            $oracionM = strtoupper($oracion);
            try {
             $stmt = $db->prepare("update oracion set oracion='$oracionM' where
codigo_oracion='$codigo'");
            $db->beginTransaction();
            $stmt->execute();
            $db->commit();
            } catch (Exception $e) {
                        $db->rollBack();
                        echo "Failed: " . $e->getMessage();
            }
            $html = $orac->tituloPresenta("Datos de la Oración","Foracion");
            $html = $html.$orac->presenta("Codigo",$codigo);
            $html = $html.$orac->presenta("Oracion",$oracion);
            $html = $html.$orac->pie("Se actualizo la oración");
            echo $html;
 }
```

Módulo de registro de tipos de palabra. -

La interface en donde se realiza este proceso se mostró en la figura 3.5.4 el código es el siguiente:

```
class sintactico{
      function FormTipoPalabra($db){
                  $tipPalabra = new formato();
                $html = $tipPalabra->titulo("Registro de tipo de palabra","Ftipop");
                $html = $html.$tipPalabra->texto("Codigo","codigo","V1","10");
                $html = $html.$tipPalabra->texto("Descripción","descrip","Verbo en primera
persona","60");
                 $ruta1="index.php?cmd=grabaTipoPalabra";
                $ruta2="index.php?cmd=listaTipoPalabra";

$graba="\"javascript:grabaTipoPala(Ftipop,'".$ruta1."','".$ruta2."','contenido','listacont')\"";
                $html = $html.$tipPalabra->boton($graba);
                echo $html;
}
    function grabaTipoPalabra($codigo,$descripcion,$db){
                  $codigoM = strtoupper($codigo);
                $tipPalabra = new formato();
                 try {
                $stmt = $db->prepare("INSERT INTO tipo_palabra
(codigo_tipo_palabra,descripcion_tipo_palabra)
                            VALUES ('$codigoM','$descripcion')");
                $db->beginTransaction();
                $stmt->execute();
                $db->commit();
              } catch (Exception $e) {
                $db->rollBack();
                echo "Failed: " . $e->getMessage();
              }
              $html = $tipPalabra->tituloPresenta("Datos del tipo de palabra","FtipoPalabra");
              $html = $html.$tipPalabra->presenta("Codigo",$codigoM);
              $html = $html.$tipPalabra->presenta("Descripcion",$descripcion);
              $html = $html.$tipPalabra->pie("Se grabo el cliente con estos datos");
              echo $html;
}

function actuaTipoPalabra($codigo,$db){
              try {
              $st = $db->prepare("select descripcion_tipo_palabra from tipo_palabra where
codigo_tipo_palabra='$codigo'");
              $st->execute();
              $result = $st->fetch(PDO::FETCH_ASSOC);
              $descrip = $result['descripcion_tipo_palabra'];
```

```
        }
        catch (Exception $e) {
                Echo "Failed: " . $e->getMessage();
        }
        $tipPalabra = new formato();
        $html = $tipPalabra->titulo("Actualiza de tipo de palabra","Ftipop");
        $html = $html.$tipPalabra->atexto("Codigo","codigo",$codigo,"10");
        $html = $html.$tipPalabra->atexto("Descripción","descrip",$descrip,"60");
        $ruta1="index.php?cmd=grabaActuaTipoPalabra";
        $ruta2="index.php?cmd=listaTipoPalabra";
       $graba="\"javascript:grabaTipoPala(Ftipop,'".$ruta1."','".$ruta2."','contenido','listacont')\
"";
        $html = $html.$tipPalabra->boton($graba);
        echo $html;
       }

       function listaTipoPalabra($db){
                $html= "<table border=0 align=center>
                <tr>
                <td colspan=3 class=titulo>Tipos de palabras</td>
                </tr>";
        try {
       $st = $db->prepare("select codigo_tipo_palabra,descripcion_tipo_palabra from
tipo_palabra");
       $st->execute();
       $html=$html."<br><br><tr>
                <td class=campo></td>
                <td class=cabCentro>Código</td>
                <td class=cabIzquierda>Descripción del tipo de palabra</td>

</tr>";
        while ($result = $st->fetch(PDO::FETCH_ASSOC)) {
                $ruta =
"index.php?cmd=actuaTipoPalabra&codigo=".$result['codigo_tipo_palabra'];
                $html=$html."<tr>
                <td class=filaDerecha>
                 href=\"#\" onclick=\"javascript:cargaRuta('".$ruta."','contenido')\">
                 <img src=images/editar.svg alt=\"Actualizar datos\" height=30 width=30
border=0 title=\"Actualizar datos\">
                </a>
                </td>
                <td class=filaCentro>".$result['codigo_tipo_palabra']."</td>
                <td class=filaIzquierda>".$result['descripcion_tipo_palabra']."</td>
                </tr>";
       }
       $html=$html."</table>";
}
catch (Exception $e) {
```

```
        Echo "Failed: " . $e->getMessage();
}
echo $html;
}
```

Módulo de asignación de tipo de palabra. -

La interface en donde se realiza este proceso se mostró en la figura 3.5.5 el código es el siguiente:

```
function AsignaTipoPalabras($db){
        $basic = new datosbasicos();
        $html= "<table border=0 align=center>
                <tr>
                <td colspan=2 class=titulo>Lista de oraciones</td>
                </tr>";
        try {
            $st = $db->prepare("select codigo_oracion,oracion from oracion");
            $st->execute();
            $html=$html."<br><br><tr>
                  <td class=cabCentro>Código</td>
                  <td class=cabIzquierda>Oración</td>
         </tr>";
        while ($result = $st->fetch(PDO::FETCH_ASSOC)) {
              $ruta = "index.php?cmd=actualizaOracion&codigo=".$result['codigo_oracion'];
              $html=$html."<tr>
                 <td class=filaCentro>".$result['codigo_oracion']."</td>
                 <td class=filaIzquierda>".$result['oracion']."</td>
                 </tr>";
                  $palabras = explode(" ", $result['oracion']);
                  $max = sizeof($palabras);
                  for ($i = 0; $i < $max; $i++){
                  try {
                       $sti = $db->prepare("select palabra from palabra where
palabra='$palabras[$i]'");
                       $sti->execute();
                       if (!($resulti = $sti->fetch(PDO::FETCH_ASSOC))){
                       $codigo = $basic->generaCodigo("palabra","codigo_palabra",$db);
                       try {
                            $stmt = $db->prepare("INSERT INTO palabra (codigo_palabra,palabra)
                                    VALUES ('$codigo','".trim($palabras[$i])."')");
                            $db->beginTransaction();
                            $stmt->execute();
                            $db->commit();
                     } catch (Exception $e) {
```

```
                    $db->rollBack();
                    echo "Failed: " . $e->getMessage();
                    }
            }

}catch (Exception $e) {
            Echo "Failed: " . $e->getMessage();
            }
            }
   }

            $html=$html."</table>";
}
catch (Exception $e) {
          Echo "Failed: " . $e->getMessage();
}
$html= $html."<table border=0 align=center>
 <tr>
 <td colspan=3 class=titulo>Lista de palabras</td>
 </tr>";
try {
    $st = $db->prepare("select codigo_palabra,palabra,codigo_tipo_palabra from palabra");
    $st->execute();
    $html=$html."<br><br><tr>
            <td class=cabCentro>Código</td>
            <td class=cabIzquierda>Palabra</td>
            <td class=cabIzquierda>Tipo palabra</td>
              </tr>";
            while ($result = $st->fetch(PDO::FETCH_ASSOC)) {
                    $codPala=$result['codigo_palabra'];
                    $html=$html."<tr>
                   <td class=filaCentro>".$codPala."</td>
                   <td class=filaIzquierda>".$result['palabra']."</td>
                   <td class=filaIzquierda><div id=".$codPala.">";
                  $tiPala=$result['codigo_tipo_palabra'];

$rutaAs="index.php?cmd=listaTipoAsigna&codigopala=".$codPala."&codigoTipo=".$tiPala;
                  $stp = $db->prepare("select descripcion_tipo_palabra from tipo_palabra where
codigo_tipo_palabra='$tiPala'");
                  $stp->execute();
                 if ($restp = $stp->fetch(PDO::FETCH_ASSOC)) {
                $html=$html.$tiPala." : ".$restp['descripcion_tipo_palabra']."<br>";
               $html=$html."<a href=# onclick=\"javascript:cargaRuta('".$rutaAs."','".$codPala."');\">
Cambiar tipo de palabra </a>";
            }
            else{
                 $html=$html.$tiPala." No tiene asignado el tipo de palabra<br>";
```

```
            $html=$html."<a href=# onclick=\"javascript:cargaRuta('".$rutaAs."','".$codPala."');\">
Seleccionar tipo de palabra </a>";
        }
        $html=$html."</div></td>
        </tr>";
        }
}
        catch (Exception $e) {
                Echo "Failed: " . $e->getMessage();
        }
         echo $html;
         }

 function ListaTipoAsigna($codigoPalabra,$codigoTipoPala,$db){
        $html="";
        try {
        $stpa = $db->prepare("select descripcion_tipo_palabra from tipo_palabra where
codigo_tipo_palabra='$codigoTipoPala'");
        $stpa->execute();
        if ($restpa = $stpa->fetch(PDO::FETCH_ASSOC)) {
         $html=$html.$codigoTipoPala." : ".$restpa['descripcion_tipo_palabra']."<br>";
         //$html=$html."<a href=# > Cambiar tipo de palabra </a>";
        }
        else{
             $html=$html." No tiene asignado el tipo de palabra<br>";
            //$html=$html."<a href=#
onclick=\"javascript:cargaRuta('".$rutaAs."','".$codPala."');\"> Seleccionar tipo de palabra </a>";
        }
        $stp = $db->prepare("select codigo_tipo_palabra,descripcion_tipo_palabra from
tipo_palabra");
        $stp->execute();
        while ($restp = $stp->fetch(PDO::FETCH_ASSOC)) {
                $tipopalaaa=$restp['descripcion_tipo_palabra'];
                $tipoPala=$restp['codigo_tipo_palabra'];

        $rutaAsig="index.php?cmd=grabaTipoAsigna&codigopala=".$codigoPalabra."&codigoTipo
        =".$tipoPala."&tipopalaa=".$tipopalaaa;
                    $html=$html."<a href=#
onclick=\"javascript:cargaRuta('".$rutaAsig."','".$codigoPalabra."');\">".$restp['codigo_tipo_palab
ra']." ".$restp['descripcion_tipo_palabra']."</a><br>";
        //$html=$html."<a
href=index.php?cmd=grabatipopalabra&codigop=".$result['codigo_palabra']."&codigotp=".$restp[
'codigo_tipo_palabra'].">".$restp['codigo_tipo_palabra']."
".$restp['descripcion_tipo_palabra']."</a><br>";
        }
}
catch (Exception $e) {
       Echo "Failed: " . $e->getMessage();
```

```
}
echo $html;
 }

 function grabaTipoPalabraP($codp,$codtp,$tipoPala,$db){
        $html="";
        $rutaAs="index.php?cmd=listaTipoAsigna&codigopala=".$codp."&codigoTipo=".$codtp;
        try {
        $stmt = $db->prepare("update palabra set codigo_tipo_palabra='$codtp' where
codigo_palabra='$codp'");
        $db->beginTransaction();
        $stmt->execute();
        $db->commit();
        } catch (Exception $e) {
        $db->rollBack();
        echo "Failed: " . $e->getMessage();
}
$html=$html.$codtp." : ".$tipoPala."<br>";
$html=$html."<a href=# onclick=\"javascript:cargaRuta('".$rutaAs."','".$codp."');\"> Cambiar tipo
de palabra </a>";
echo $html;
 }
```

Módulo de la Generación de las estructuras. -

La interface en donde se realiza este proceso se mostró en la figura 3.5.6 el código es el siguiente:

```
function generaEstructura($db){
        $basic = new datosbasicos();
        $estruct = "";
        $estr_num = "";
        $html= "<table border=0 align=center>
         <tr>
         <td colspan=4 class=titulo>Lista de oraciones</td>
         </tr>";
        try {
            $st = $db->prepare("select codigo_oracion,oracion from oracion");
            $st->execute();
            $html=$html."<br><br><tr>
           <td class=cabCentro>Código</td>
           <td class=cabIzquierda>Oración</td>
           <td class=cabIzquierda>Estructura</td>
           <td class=cabIzquierda>Estructura numérica</td>
         </tr>";
         while ($result = $st->fetch(PDO::FETCH_ASSOC)) {
```

```
                $codOra = $result['codigo_oracion'];
                $estruct = "";
               $estr_num = "";
                $palabras = explode(" ", $result['oracion']);
              $max = sizeof($palabras);
             for ($i = 0; $i < $max; $i++){
              try {
                 $sti = $db->prepare("select a.codigo_palabra, a.palabra, a.codigo_tipo_palabra,
b.numero_tipo_palabra from palabra a, tipo_palabra b where a.palabra='$palabras[$i]' and
a.codigo_tipo_palabra=b.codigo_tipo_palabra");
                $sti->execute();
                if ($resulti = $sti->fetch(PDO::FETCH_ASSOC)){
                 $estruct = $estruct." ".$resulti['codigo_tipo_palabra'];
                 $estr_num = $estr_num." ".$resulti['numero_tipo_palabra'];
                 }
            }catch (Exception $e) {
                  Echo "Failed: " . $e->getMessage();
             }
        }
        $estruct = trim($estruct);
        try {
        $stmt = $db->prepare("update oracion set estructura='$estruct',
estructura_numerica='$estr_num' where codigo_oracion='$codOra'");
        $db->beginTransaction();
        $stmt->execute();
        $db->commit();
        } catch (Exception $e) {
        $db->rollBack();
        echo "Failed: " . $e->getMessage();
        }
        $html=$html."<tr>
        <td class=filaCentro>".$codOra."</td>
        <td class=filaIzquierda>".$result['oracion']."</td>
        <td class=filaIzquierda>".$estruct."</td>
        <td class=filaIzquierda>".$estr_num."</td>
</tr>";
 $palabras = explode(" ", $result['oracion']);
 $max = sizeof($palabras);
 }
$html=$html."</table>";
}
catch (Exception $e) {
      Echo "Failed: " . $e->getMessage();
}
echo $html;
 }
```

Módulo del Aprendizaje de la RNA. -

La interface en donde se realiza este proceso se mostró en la figura 3.5.2 y el código es el siguiente:

```
function aprendizajeRed($db){
        $basic = new datosbasicos();
        $cRNA = new libRNA();
        $capa=0;
        $numNeuro=0;
        $numEntra=0;
        $html= "<table border=0 align=center>
         <tr>
         <td colspan=4 class=titulo>Estructura de Red Neuronal Artificial</td>
         </tr>";
try {
 $st = $db->prepare("select id_estructura_neurona, dato_estructura_neurona from
estructura_neurona");
 $st->execute();
$html=$html."<br><br><tr>
<td class=cabCentro>Estructura</td>
<td class=cabIzquierda>Dato</td>
</tr>";
while ($result = $st->fetch(PDO::FETCH_ASSOC)) {
       $idEstruc = $result['id_estructura_neurona'];
       $datoEstruct = $result['dato_estructura_neurona'];
       $EstrucRNA[$idEstruc-1] = $datoEstruct;
      /*$html=$html."<tr>
       <td class=filaCentro>".$idEstruc."</td>
      <td class=filaIzquierda>".$datoEstruct."</td>
      </tr>";*/
}
for ($i=0; $i<=$EstrucRNA[0]; $i++){
$html=$html."<tr>
 <td class=filaCentro>".$i."</td>
 <td class=filaIzquierda>".$EstrucRNA[$i]."</td>
 </tr>";
}
try {
    $stneuro = $db->prepare("select id_neurona, u, k, a, n, capa, m from neurona");
    $stneuro->execute();
    $numNeuro=0;
    while ($rneuro = $stneuro->fetch(PDO::FETCH_ASSOC)) {
        $numNeuro++;
        $m[$rneuro['id_neurona']] = $rneuro['m'];
        $caparna[$rneuro['id_neurona']] = $rneuro['capa'];
```

```
            $n[$rneuro['id_neurona']] = $rneuro['n'];
            $a[$rneuro['id_neurona']] = $rneuro['a'];
            $k[$rneuro['id_neurona']] = $rneuro['k'];
            $u[$rneuro['id_neurona']] = $rneuro['u'];
            try {
            $stpeso = $db->prepare("select id_neurona,id_peso,valor_peso from peso order by
id_neurona,id_peso");
            $stpeso->execute();
            while ($rpeso = $stpeso->fetch(PDO::FETCH_ASSOC)) {
            $pesorna[$rpeso['id_neurona']][$rpeso['id_peso']] = $rpeso['valor_peso'];
            }
             }catch (Exception $e) {
                    Echo "Failed: " . $e->getMessage();
             }
}
}
catch (Exception $e) {
        Echo "Failed: " . $e->getMessage();
}
$html=$html."<tr>
<td colspan=4 class=titulo>Red Neuronal Artificial $numNeuro</td>
</tr>";
/*for ($ineuro=1; $ineuro <= $numNeuro; $ineuro++){
$html=$html."<tr>
 <td class=filaCentro>Neurona ".$ineuro."</td>
 <td class=filaIzquierda>
 Capa : ".$caparna[$ineuro]."<br>
 Entradas : ".$m[$ineuro]."<br>
 Sumatoria : ".$n[$ineuro]."<br>
 Resultado : ".$a[$ineuro]."<br>
 Umbral : ".$u[$ineuro]."<br>
 K adaline : ".$k[$ineuro]."<br>";
 $ne = $m[$ineuro];
 for ($npes=1; $npes <= $ne; $npes++){
      $html=$html."peso[".$npes."] : ".$pesorna[$ineuro][$npes]."<br>";
 }
$html=$html."    </td>
 </tr>";
}*/
/*****************************  APRENDIZAJE  *******************************/

$html=$html."<tr>
 <td colspan=4 class=titulo>Aprendizaje Red Neuronal Artificial</td>
 </tr>";
try {
     $xMax = 1600;
     $yMax = 400;
     $xc1 = 40;
```

```
    $yc1 = $yMax/2;
    $xzg = $xMax - 80;
    $yzg = $yMax/2 - 20;
    $xValMax = 20000000;
    $yValMax = 1000;
    $ex = $xzg/$xValMax;
   $ey = $yzg/$yValMax;
   $im = imagecreatetruecolor($xMax, $yMax);
   $text_color = imagecolorallocate($im, 233, 14, 91);
   $line_color = imagecolorallocate($im, 0, 0, 0);
   $resul_color = imagecolorallocate($im, 255, 0, 0);
   $umbral_color = imagecolorallocate($im, 0, 0, 255);
   $itera_color = imagecolorallocate($im, 0, 255, 0);
   $bg_Color = imagecolorallocate($im, 255, 255, 255);
   imagefilledrectangle($im,0,0,$xMax, $yMax,$line_color);
   imagefilledrectangle($im,1,1,$xMax-2, $yMax-2,$bg_Color);
  imagestring($im, 5,$xzg/2-10*17, 5,  'Evolución del aprendizaje de la Red Neuronal Artificial',
$line_color);
  imageline ($im, $xc1, $yc1, $xc1+$xzg, $yc1,$line_color);
  imageline ($im, $xc1, $yc1-$yzg, $xc1, $yc1+$yzg,$line_color);
  $ixcoor = $xzg/10;
  $ivxmax = $xValMax/10;
  for ($ix=1; $ix<=10; $ix++){
      imageline ($im, $xc1+$ix*$ixcoor, $yc1+5, $xc1+$ix*$ixcoor, $yc1,$line_color);
      imagestring($im, 3, $xc1+$ix*$ixcoor-30, $yc1+10,$ix*$ivxmax, $line_color);
}
$iycoor = $yzg/5;
$ivymax = $yValMax/5;
for ($iy=1; $iy<=5; $iy++){
    imageline ($im, $xc1, $yc1-$iy*$iycoor, $xc1-5, $yc1-$iy*$iycoor,$line_color);
    imageline ($im, $xc1, $yc1+$iy*$iycoor, $xc1-5, $yc1+$iy*$iycoor,$line_color);
    imagestring($im, 3, $xc1-35, $yc1+$iy*$iycoor-8,$iy*$ivymax, $line_color);
    imagestring($im, 3, $xc1-35, $yc1-$iy*$iycoor-8,$iy*$ivymax, $line_color);
}
imageline ($im,$xc1+$ex*$cActuaPesosa,$yc1-$ey*$u[$numNeuro],$xc1+$xzg,$yc1-
$ey*$u[$numNeuro],$umbral_color);
                                    $ci = 10000;
                                    $ineurona = 1;
                                    $ka = 0.0000001;
                                    $cActuaPesos = 0;
                                    $cActuaPesosa = 0;
                                    $resRNA = 0;
                                    $resRNAa = 0;
                                    $html=$html."<tr>
          <td class=filaCentro>Estructura </td>
          <td class=filaIzquierda>";
         for ($iit=0; $iit<$ci; $iit++){
```

```
$html=$html." ************************* Iteracion :
".$iit."*******************************<br>";

$stEstruc = $db->prepare("select estructura_numerica from oracion order by codigo_oracion limit
10");
$stEstruc->execute();
while ($rEstruc = $stEstruc->fetch(PDO::FETCH_ASSOC)) {
        $Estruc = trim($rEstruc['estructura_numerica']);
        $html=$html." Oracion : ".$Estruc."<br>";
        $entrada = explode(" ",$Estruc);
        $max = sizeof($entrada);
        for ($i=$max; $i<10; $i++){
            $entrada[$i]="0";
 }
            /*for ($i=0; $i<10; $i++){
            $html=$html." entrada[".$i."] : ".$entrada[$i]."<br>";
            }*/
$resRNA = $cRNA-
>calculoRNA($entrada,$n,$m,$caparna,$a,$k,$u,$pesorna,$numNeuro,$EstrucRNA);
$html=$html." Valor incial RNA  : ".$resRNA."<br>";
$ineurona=1;
do{
 $resRNA = $cRNA-
>calculoRNA($entrada,$n,$m,$caparna,$a,$k,$u,$pesorna,$numNeuro,$EstrucRNA);
 $AP = $u[$numNeuro]-$resRNA;
 if ($caparna[$ineurona]==1){
     if ($AP>0){
        if ($AP>1000){
           $AP=1000;
        }
       for ($ip=1; $ip<=$m[$ineurona]; $ip++){
            $cActuaPesos++;
            $pesorna[$ineurona][$ip] = $pesorna[$ineurona][$ip] + $ka*$AP*$entrada[$ip];
       }
}
       else{
            if ($AP<-1000){
           $AP=-1000;
            }
            for ($ip=1; $ip<=$m[$ineurona]; $ip++){
            $cActuaPesos++;
            $pesorna[$ineurona][$ip] = $pesorna[$ineurona][$ip] + $ka*$AP*$entrada[$ip];
            }
             }
}
else{
       if ($AP>0){
          if ($AP>1000){
```

```
            $AP=1000;
        }
  for ($ip=1; $ip<=$m[$ineurona]; $ip++){
        $cActuaPesos++;
        $pesorna[$ineurona][$ip] = $pesorna[$ineurona][$ip] + $ka*$AP;
}
 }
 else{
       if ($AP<-1000){
          $AP=-1000;
        }
       for ($ip=1; $ip<=$m[$ineurona]; $ip++){
            $cActuaPesos++;
            $pesorna[$ineurona][$ip] = $pesorna[$ineurona][$ip] + $ka*$AP;
}
 }
}
if ($ineurona<$numNeuro){
   $ineurona++;
}
else{
      $ineurona=1;
     }
imageline ($im,$xc1+$ex*$cActuaPesosa,$yc1-$ey*$resRNAa,$xc1+$ex*$cActuaPesos,$yc1-
$ey*$resRNA,$resul_color);
$cActuaPesosa=$cActuaPesos;
$resRNAa = $resRNA;
}while(abs($u[$numNeuro]-$resRNA)>20);

$html=$html." Valor final RNA  : ".$resRNA."<br>";
}
imageline ($im,$xc1+$ex*$cActuaPesos,$yc1+$yzg,$xc1+$ex*$cActuaPesos,$yc1-
$yzg,$itera_color);
}
imagejpeg($im, 'images/aprendeRNA.jpg');
imagedestroy($im);
$html=$html." Cantidad actualizaciones : ".$cActuaPesos."<br>";
$html=$html." Evolucion del aprendizaje de la RNA : <br><img src=images/aprendeRNA.jpg><br>";
$html=$html."    </td>
 </tr>";
}catch (Exception $e) {
 Echo "Failed: " . $e->getMessage();
}
for ($ineuro=1; $ineuro <= $numNeuro; $ineuro++){
/*$html=$html."<tr>
 <td class=filaCentro>Neurona ".$ineuro."</td>
 <td class=filaIzquierda>
 Capa : ".$caparna[$ineuro]."<br>
```

```
 Entradas : ".$m[$ineuro]."<br>
 Sumatoria : ".$n[$ineuro]."<br>
 Resultado : ".$a[$ineuro]."<br>
 Umbral : ".$u[$ineuro]."<br>
 K adaline : ".$k[$ineuro]."<br>";*/
$ne = $m[$ineuro];
for ($npes=1; $npes <= $ne; $npes++){
//$html=$html."pesorna[".$ineuro."][".$npes."] : ".$pesorna[$ineuro][$npes]."<br>";
try {
  //$peso=rand(0, 10000) / 10000;
    $pesoVal=$pesorna[$ineuro][$npes];
    $udpeso = $db->prepare("update peso set valor_peso='$pesoVal' where id_peso='$npes' and
id_neurona='$ineuro'");
$db->beginTransaction();
$udpeso->execute();
$db->commit();
//$pesorna[$numNeuro][$j] = $peso;
} catch (Exception $e) {
 $db->rollBack();
 echo "Failed: " . $e->getMessage();
          }
}
/*$html=$html."            </td>
 </tr>";*/
}
$html=$html."</table>";
}
catch (Exception $e) {
        Echo "Failed: " . $e->getMessage();
}
echo $html;
}
```

Modulo para generar red neuronal. -

Este módulo inicializa la RNA en la base de datos, el código se muestra a continuación y el resultado de ejecución del módulo ver en el anexo 1.

```
function generaRed($db){
          $basic = new datosbasicos();
          $capa=0;
          $numNeuro=0;
          $numEntra=0;
          $html= "<table border=0 align=center>
           <tr>
           <td colspan=4 class=titulo>Estructura de Red Neuronal Artificial</td>
           </tr>";
```

```
        try {
        $st = $db->prepare("select id_estructura_neurona, dato_estructura_neurona from
estructura_neurona");
         $st->execute();
        $html=$html."<br><br><tr>
        <td class=cabCentro>Estructura</td>
        <td class=cabIzquierda>Dato</td>
        </tr>";
        while ($result = $st->fetch(PDO::FETCH_ASSOC)) {
                $idEstruc = $result['id_estructura_neurona'];
                $datoEstruct = $result['dato_estructura_neurona'];
                if ($idEstruc==1){
                    $numero_capas = $datoEstruct;
                }
               if ($idEstruc==2){
                  $capa++;
                  $numEntra=10;
                 for ($i=1; $i<=$datoEstruct; $i++){
                   $numNeuro++;
                    try {
                   $stmt = $db->prepare("insert into neurona (id_neurona,m,capa,n,a,k,u)
values('$numNeuro','$numEntra','$capa','0','0','1','500')");
                   $db->beginTransaction();
                   $stmt->execute();
                   $db->commit();
                   $m[$numNeuro] = $numEntra;
                  $caparna[$numNeuro] = $capa;
                  $n[$numNeuro] = 0;
                  $a[$numNeuro] = 0;
                  $k[$numNeuro] = 1;
                  $u[$numNeuro] = 500;
                  for ($j=1; $j<=$numEntra; $j++){
                   try {
                       $peso = rand(0, 10000) / 10000;
                       $stpeso = $db->prepare("insert into peso (id_peso,id_neurona,valor_peso)
values('$j','$numNeuro','$peso')");
                   $db->beginTransaction();
                   $stpeso->execute();
                   $db->commit();
                   $pesorna[$numNeuro][$j] = $peso;
                    } catch (Exception $e) {
                            $db->rollBack();
                            echo "Failed: " . $e->getMessage();
                   }
         }
        } catch (Exception $e) {
                   $db->rollBack();
                   echo "Failed: " . $e->getMessage();
```

```
        }
        }
        $numEntra=$datoEstruct;
       }
       if ($idEstruc>2){
         $capa++;
         for ($i=1; $i<=$datoEstruct; $i++){
              $numNeuro++;
              try {
                   $stmt = $db->prepare("insert into neurona (id_neurona,m,capa,n,a,k,u)
values('$numNeuro','$numEntra','$capa','0','0','1','500')");
                  $db->beginTransaction();
                  $stmt->execute();
                  $db->commit();
                  $m[$numNeuro] = $numEntra;
                  $caparna[$numNeuro] = $capa;
                  $n[$numNeuro] = 0;
                  $a[$numNeuro] = 0;
                  $k[$numNeuro] = 1;
                  $u[$numNeuro] = 500;
                  for ($j=1; $j<=$numEntra; $j++){
                  try {
                   $peso=rand(0, 10000) / 10000;
                   $stpeso = $db->prepare("insert into peso (id_peso,id_neurona,valor_peso)
values('$j','$numNeuro','$peso')");
                   $db->beginTransaction();
                   $stpeso->execute();
                   $db->commit();
                   $pesorna[$numNeuro][$j] = $peso;
                   } catch (Exception $e) {
                          $db->rollBack();
                          echo "Failed: " . $e->getMessage();
                   }
        }
        } catch (Exception $e) {
                   $db->rollBack();
                   echo "Failed: " . $e->getMessage();
        }
        }
        $numEntra=$datoEstruct;
       }
       $html=$html."<tr>
       <td class=filaCentro>".$idEstruc."</td>
       <td class=filaIzquierda>".$datoEstruct."</td>
       </tr>";
       }
       $html=$html."<tr>
        <td colspan=4 class=titulo>Red Neuronal Artificial $numNeuro</td>
```

```
</tr>";
for ($ineuro=1; $ineuro <= $numNeuro; $ineuro++){
   $html=$html."<tr>
   <td class=filaCentro>Neurona ".$ineuro."</td>
   <td class=filaIzquierda>
    Capa : ".$caparna[$ineuro]."<br>
 Entradas : ".$m[$ineuro]."<br>
 Sumatoria : ".$n[$ineuro]."<br>
 Resultado : ".$a[$ineuro]."<br>
 Umbral : ".$u[$ineuro]."<br>
 K adaline : ".$k[$ineuro]."<br>";
 $ne = $m[$ineuro];
 for ($npes=1; $npes <= $ne; $npes++){
 $html=$html."peso[".$npes."] : ".$pesorna[$ineuro][$npes]."<br>";
 }
$html=$html."    </td>
 </tr>";
}
$html=$html."</table>";
}
catch (Exception $e) {
 Echo "Failed: " . $e->getMessage();
}
echo $html;
            }
```

El diseño de la estructura de la RNA para el analizador sintáctico, corresponde a una red neuronal artificial de pre alimentación con neuronas de activación lineal salvo la última capa que tiene activación hardlim y que consta de cinco capas ver figura 3.5.9, donde en la primera capa tenemos diez (10) neuronas, en la segunda capa tiene quince (15) neuronas, en la tercera capa diez (10) neuronas, en la cuarta capa tiene ocho (8) y en la quinta y última capa una (1) neurona.

La red tiene 10 entradas que equivalen a la cantidad máxima de palabras de una oración.

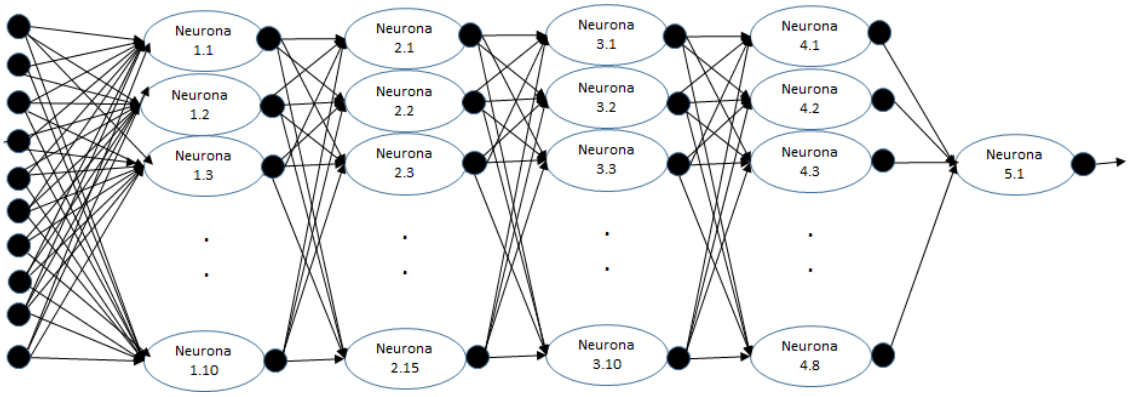


Figura 3.5.9 Estructura de la RNA.

Módulo de reconocimiento de la estructura. -

Realiza el reconocimiento de la estructura sintáctica, el código se muestra a continuación.

```
function ReconocimientoEstruct($db){
        $basic = new datosbasicos();
        $cRNA = new libRNA();
        $capa=0;
        $numNeuro=0;
        $numEntra=0;
        $html= "<table border=0 align=center>
         <tr>
         <td colspan=4 class=titulo>Estructura de Red Neuronal Artificial</td>
         </tr>";
         try {
         $st = $db->prepare("select id_estructura_neurona, dato_estructura_neurona from
estructura_neurona");
        $st->execute();
        $html=$html."<br><br><tr>
        <td class=cabCentro>Estructura</td>
        <td class=cabIzquierda>Dato</td>
          </tr>";
        while ($result = $st->fetch(PDO::FETCH_ASSOC)) {
                $idEstruc = $result['id_estructura_neurona'];
                $datoEstruct = $result['dato_estructura_neurona'];
                $EstrucRNA[$idEstruc-1] = $datoEstruct;
                /*$html=$html."<tr>
                <td class=filaCentro>".$idEstruc."</td>
                <td class=filaIzquierda>".$datoEstruct."</td>
                </tr>";*/
```

```
        }
        for ($i=0; $i<=$EstrucRNA[0]; $i++){
            $html=$html."<tr>
            <td class=filaCentro>".$i."</td>
            <td class=filaIzquierda>".$EstrucRNA[$i]."</td>
            </tr>";
        }
        try {
            $stneuro = $db->prepare("select id_neurona, u, k, a, n, capa, m from neurona");
            $stneuro->execute();
            $numNeuro=0;
           while ($rneuro = $stneuro->fetch(PDO::FETCH_ASSOC)) {
                  $numNeuro++;
                  $m[$rneuro['id_neurona']] = $rneuro['m'];
                  $caparna[$rneuro['id_neurona']] = $rneuro['capa'];
                  $n[$rneuro['id_neurona']] = $rneuro['n'];
                  $a[$rneuro['id_neurona']] = $rneuro['a'];
                  $k[$rneuro['id_neurona']] = $rneuro['k'];
                  $u[$rneuro['id_neurona']] = $rneuro['u'];
                  try {
                      $stpeso = $db->prepare("select id_neurona,id_peso,valor_peso from peso
order by id_neurona,id_peso");
                     $stpeso->execute();
                     while ($rpeso = $stpeso->fetch(PDO::FETCH_ASSOC)) {
                           $pesorna[$rpeso['id_neurona']][$rpeso['id_peso']] = $rpeso['valor_peso'];
                     }
                  }catch (Exception $e) {
                           Echo "Failed: " . $e->getMessage();
                  }
            }
            }
            catch (Exception $e) {
                  Echo "Failed: " . $e->getMessage();
        }
            $html=$html."<tr>
            <td colspan=4 class=titulo>Red Neuronal Artificial $numNeuro</td>
            </tr>";
        /*for ($ineuro=1; $ineuro <= $numNeuro; $ineuro++){
            $html=$html."<tr>
            <td class=filaCentro>Neurona ".$ineuro."</td>
            <td class=filaIzquierda>
            Capa : ".$caparna[$ineuro]."<br>
            Entradas : ".$m[$ineuro]."<br>
           Sumatoria : ".$n[$ineuro]."<br>
           Resultado : ".$a[$ineuro]."<br>
           Umbral : ".$u[$ineuro]."<br>
           K adaline : ".$k[$ineuro]."<br>";
           $ne = $m[$ineuro];
```

```
        for ($npes=1; $npes <= $ne; $npes++){
            $html=$html."peso[".$npes."] : ".$pesorna[$ineuro][$npes]."<br>";
        }
        $html=$html."    </td>
        </tr>";
        }*/

        /***************************** APRENDIZAJE ***************************/
                $html=$html."<tr>
         <td colspan=4 class=titulo>Reconocimiento de estructuras con la Red Neuronal
Artificial</td>
         </tr>";
        try {
           /*$xMax = 1600;
           $yMax = 400;
           $xc1 = 40;
           $yc1 = $yMax/2;
           $xzg = $xMax - 80;
           $yzg = $yMax/2 - 20;
           $xValMax = 20000000;
           $yValMax = 1000;
           $ex = $xzg/$xValMax;
           $ey = $yzg/$yValMax;
           $im = imagecreatetruecolor($xMax, $yMax);
           $text_color = imagecolorallocate($im, 233, 14, 91);
           $line_color = imagecolorallocate($im, 0, 0, 0);
           $resul_color = imagecolorallocate($im, 255, 0, 0);
           $umbral_color = imagecolorallocate($im, 0, 0, 255);
           $itera_color = imagecolorallocate($im, 0, 255, 0);
           $bg_Color = imagecolorallocate($im, 255, 255, 255);
           imagefilledrectangle($im,0,0,$xMax, $yMax,$line_color);
           imagefilledrectangle($im,1,1,$xMax-2, $yMax-2,$bg_Color);
           imagestring($im, 5,$xzg/2-10*17, 5,  'Evolución del aprendizaje de la Red Neuronal
Artificial', $line_color);
          imageline ($im, $xc1, $yc1, $xc1+$xzg, $yc1,$line_color);
          imageline ($im, $xc1, $yc1-$yzg, $xc1, $yc1+$yzg,$line_color);
          $ixcoor = $xzg/10;
          $ivxmax = $xValMax/10;
          for ($ix=1; $ix<=10; $ix++){
              imageline ($im, $xc1+$ix*$ixcoor, $yc1+5, $xc1+$ix*$ixcoor, $yc1,$line_color);
             imagestring($im, 3, $xc1+$ix*$ixcoor-30, $yc1+10,$ix*$ivxmax, $line_color);
          }
        $iycoor = $yzg/5;
        $ivymax = $yValMax/5;
        for ($iy=1; $iy<=5; $iy++){
            imageline ($im, $xc1, $yc1-$iy*$iycoor, $xc1-5, $yc1-$iy*$iycoor,$line_color);
            imageline ($im, $xc1, $yc1+$iy*$iycoor, $xc1-5, $yc1+$iy*$iycoor,$line_color);
            imagestring($im, 3, $xc1-35, $yc1+$iy*$iycoor-8,$iy*$ivymax, $line_color);
```

```
            imagestring($im, 3, $xc1-35, $yc1-$iy*$iycoor-8,$iy*$ivymax, $line_color);
        }
        imageline ($im,$xc1+$ex*$cActuaPesosa,$yc1-$ey*$u[$numNeuro],$xc1+$xzg,$yc1-
$ey*$u[$numNeuro],$umbral_color);*/
        $ci = 10000;
        $ineurona = 1;
        $ka = 0.0000001;
        $cActuaPesos = 0;
        $cActuaPesosa = 0;
        $resRNA = 0;
        $resRNAa = 0;
        $html=$html."<tr>
         <td class=filaCentro>Estructura </td>
         <td class=filaIzquierda>";
        //for ($iit=0; $iit<$ci; $iit++){
//$html=$html." ********************** Iteracion
:".$iit."*****************************<br>";
$stEstruc = $db->prepare("select oracion,estructura_numerica from oracion order by
codigo_oracion limit 10");
        $stEstruc->execute();
        while ($rEstruc = $stEstruc->fetch(PDO::FETCH_ASSOC)) {
        $Oracion = trim($rEstruc['oracion']);
        $Estruc = trim($rEstruc['estructura_numerica']);
        $html=$html." Oracion : ".$Estruc."<br>";
        $html=$html." Oracion : ".$Oracion."<br>";
         $entrada = explode(" ",$Estruc);
         $max = sizeof($entrada);
         for ($i=$max; $i<10; $i++){
             $entrada[$i]="0";
         }
         /*for ($i=0; $i<10; $i++){
         $html=$html." entrada[".$i."] : ".$entrada[$i]."<br>";
         }*/
$resRNA = $cRNA-
>calculoRNA($entrada,$n,$m,$caparna,$a,$k,$u,$pesorna,$numNeuro,$EstrucRNA);
$html=$html." Valor RNA  : ".$resRNA."<br>";
}
//imageline ($im,$xc1+$ex*$cActuaPesos,$yc1+$yzg,$xc1+$ex*$cActuaPesos,$yc1-
$yzg,$itera_color);
//}
//imagejpeg($im, 'images/aprendeRNA.jpg');
//imagedestroy($im);
//$html=$html." Cantidad actualizaciones : ".$cActuaPesos."<br>";
//$html=$html." Evolucion del aprendizaje de la RNA : <br><img
src=images/aprendeRNA.jpg><br>";
$html=$html."    </td>
 </tr>";
}catch (Exception $e) {
```

```
        Echo "Failed: " . $e->getMessage();
}
$html=$html."</table>";
}
catch (Exception $e) {
       Echo "Failed: " . $e->getMessage();
}
echo $html;
 }
}
```

3.5.4 Algoritmo de aprendizaje de la red neuronal artificial para la realización del análisis sintáctico. -

Para la actualización se usa el algoritmo de Backpropagation modificado, en el cual en lugar de usar el error medio cuadrático se usa el error instantáneo de la RNA. En base a esto para actualizar los pesos se usa la fórmula 3.5.10:

$$\mathbf{peso_{j,i} = peso_{j,i-1} + error * ka * entrada_j} \quad (3.5.10)$$

Donde:

$peso_{j,i}$ – Valor del peso j actual.

$peso_{j,i}$ – Valor del peso j anterior.

error = Valor deseado de salida de la RNA - salida real de la RNA

ka – Velocidad de aprendizaje.

$Entrada_j$ – Entrada de la cual se actualiza el peso.

3.5.5 Desarrollo y pruebas del software del algoritmo de aprendizaje de la red neuronal artificial para la realización del análisis sintáctico. –

Se realizó un programa en PHP para el aprendizaje de la red neuronal artificial, y se realizó las pruebas correspondientes para determinar la convergencia del algoritmo como se muestra de en las figuras 3.5.11 y 3.5.12

y en el anexo 2 se muestra el proceso completo de aprendizaje para 10 estructuras en 2000 iteraciones.

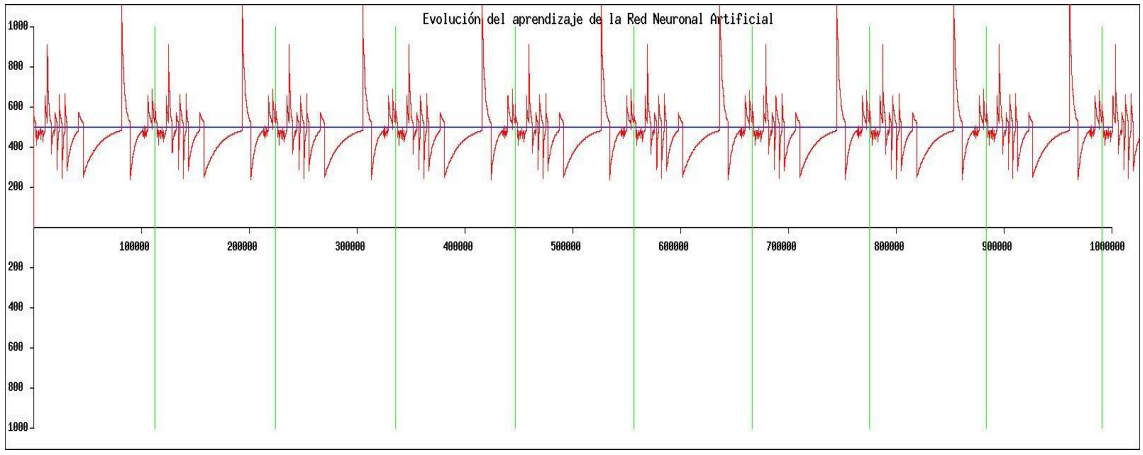


Figura 3.5.11 Evolución del aprendizaje de 10 estructuras sintácticas en 9 iteraciones

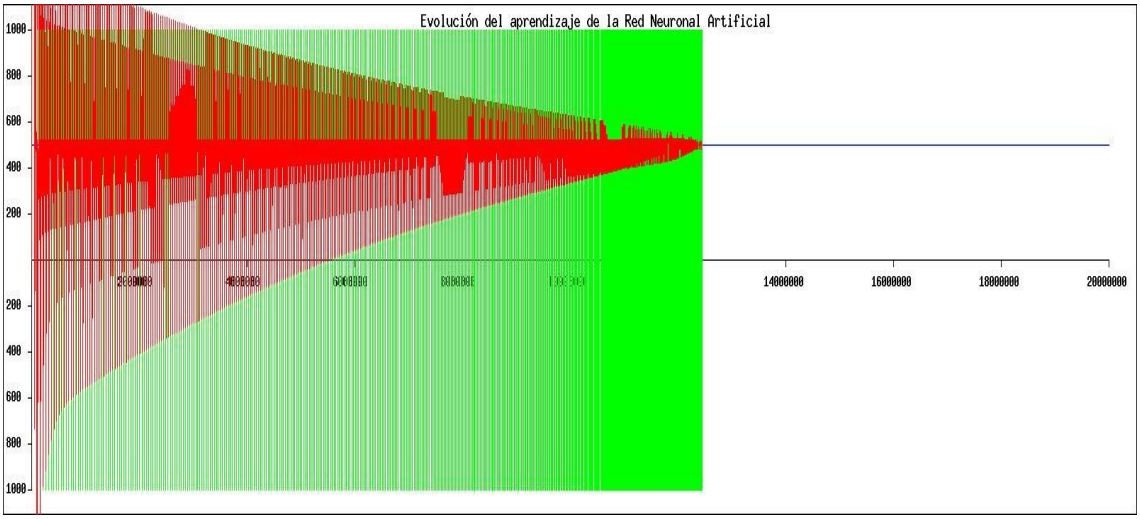


Figura 3.5.12 Evolución del aprendizaje de 10 estructuras sintácticas en 2000 iteraciones

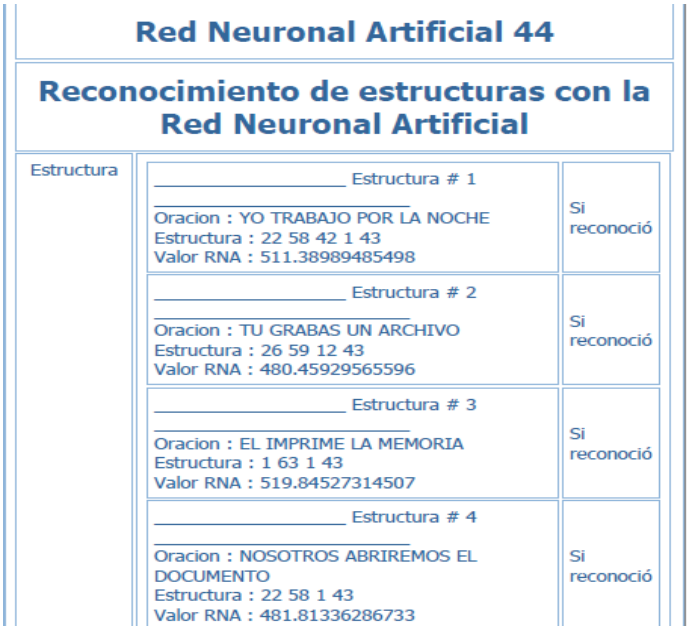
3.5.6 Desarrollo y pruebas del software del algoritmo de funcionamiento de la red neuronal artificial para la realización del análisis sintáctico. –

Se programó en PHP el algoritmo de funcionamiento de la red neuronal artificial y se realizaron las pruebas correspondientes.

3.5.7 Prueba de la eficacia del software de reconocimiento de las estructuras sintácticas. –

Se realizó con el software una serie de pruebas con diversas estructuras como se muestra en la tabla 3.5.1.

Tabla 3.5.1.



Red Neuronal Artificial 44

Reconocimiento de estructuras con la Red Neuronal Artificial

Estructura		
	_____ Estructura # 1 _____ Oracion : YO TRABAJO POR LA NOCHE Estructura : 22 58 42 1 43 Valor RNA : 511.38989485498	Si reconoció
	_____ Estructura # 2 _____ Oracion : TU GRABAS UN ARCHIVO Estructura : 26 59 12 43 Valor RNA : 480.45929565596	Si reconoció
	_____ Estructura # 3 _____ Oracion : EL IMPRIME LA MEMORIA Estructura : 1 63 1 43 Valor RNA : 519.84527314507	Si reconoció
	_____ Estructura # 4 _____ Oracion : NOSOTROS ABRIREMOS EL DOCUMENTO Estructura : 22 58 1 43 Valor RNA : 481.81336286733	Si reconoció

De la tabla 3.5.1. se puede apreciar que la eficacia en el reconocimiento de las estructuras sintacticas es del 100%.

3.6 ANALISIS DE DATOS

El análisis de datos se realizó mediante una observación de datos de mediciones obtenidas a través del prototipo y no requiere de ningún método estadístico ya que la investigación es determinista y no incluye ninguna variable estocástica.

3.7 CONSIDERACIONES ETICAS

Existen países donde, con matices, el software es patentable. El software no es patentable en Europa ni en América. Existe una gran presión para revertir esta situación por parte de la industria y por parte de grupos de presión con intereses comerciales y de las empresas cuyos únicos activos son patentes. En 2005 se rechazó en el Parlamento Europeo una propuesta para permitir las patentes. La OEPM y la OEP admiten, de facto, patentes de software El software en sí no es patentable Pero sí cuando posee un efecto técnico.

3.7.1 Riesgos de la patente de software para los fabricantes

Abre la puerta a la posibilidad de patentar ideas y algoritmos pudiendo impedir la reingeniería y la ingeniería inversa, así como también puede impedir el desarrollo de nuevo software similar y de software de interfaz, por otro lado, puede impedir, de facto, el empleo de determinados formatos y así encarecer los costes de desarrollo, además, su duración es muy elevada en un área tan dinámica.

En conclusión, las patentes de software pueden frenar la innovación, en vez de fomentarla, y perjudican a las empresas pequeñas frente a las grandes.

3.7.2 Riesgos para los usuarios

Las licencias no garantizan que no se infrinjan patentes sería difícil de determinar, la infracción podría ser indirecta y se asumiría la responsabilidad.

La finalidad de una demanda suele ser económica es decir obligar al pago de una licencia de patente cuyo objetivo serían las organizaciones

En conclusión, Un usuario puede ser demandando por infringir un derecho de patente por el mero hecho de usar un determinado software legalmente adquirido que lo haga.

IV. RESULTADOS

4.1 CONSTRASTACION DE LA HIPOTESIS

En base a las pruebas experimentales en el prototipo desarrollado e implementado donde se ha podido verificar que, si el aprendizaje la red neuronal artificial se realiza con la cantidad de muestras adecuada y además si se toma las muestras correctas, entonces el sistema tiene un comportamiento estable y puede realizar el reconocimiento de las estructuras sintácticas correspondientes. En los resultados de la tabla 2.6.1 se obtiene una precisión del 100% con todas las muestras (estructuras sintácticas) de entrenamiento. Por lo que la hipótesis "Con el uso una red neuronal artificial se podría realizar el análisis sintáctico del lenguaje natural para el idioma castellano." queda demostrada.

4.2 ANALISIS E INTERPRETACION

Analizando los resultados obtenidos podemos ver que la propuesta de la presente investigación puede aplicarse para implementar sistemas que requieran realizar el reconocimiento de las estructuras sintácticas.

V. DISCUSION DE RESULTADOS

En la presente investigación se ha desarrollado la base teórica, algoritmos y programas computacionales que permite realizar el reconocimiento de las estructuras sintácticas basados en la teoría de redes neuronales artificiales.

Para realizar el aprendizaje de la red neuronal artificial se ingresó oraciones gramaticales del idioma castellano, las cuales fueron convertidas a estructuras gramaticales de acuerdo a la formalización que se hizo del idioma castellano para el procesamiento.

Estas estructuras gramaticales sirvieron como entrada para la red neuronal artificial de la figura 3.5.9 y realizando el proceso de aprendizaje con 10 oraciones de diferente estructura gramatical se ajustaron los pesos de la red neuronal artificial en mil quinientas iteraciones haciendo un total de aproximadamente 20 millones de actualizaciones de los pesos, se obtuvo la convergencia como se muestra en las figuras 3.5.11 y 3.5.12 donde se presenta la evolución del proceso de aprendizaje.

Con el conocimiento adquirido en el proceso de aprendizaje se realizaron pruebas de reconocimiento de las oraciones ingresadas y se obtuvo un reconocimiento del 100% de las oraciones con las 10 estructuras sintácticas con las que se entrenó la red neuronal artificial.

Para incrementar el número de estructuras gramaticales diferentes se debe simplemente entrenar la red neuronal artificial con todas las estructuras gramaticales que deseamos que el sistema reconozca, lo que si es recomendable utilizar una computadora de alto rendimiento ya que el aumento de estructuras gramaticales diferentes aumenta el tiempo de procesamiento.

Como podemos ver de lo expuesto en los párrafos anteriores se logró un software que solo es necesario entrenar con las estructuras gramaticales que son necesarias y puede reconocerlas para cualquier tipo de aplicación.

VI. CONCLUSIONES

En la presente investigación podemos concluir que:

- Se ha podido identificar, categorizar y codificar las diferentes palabras del idioma castellano para su procesamiento.
- Se ha podido realizar el análisis léxico del idioma castellano a partir de una base de datos.
- Aplicando redes neuronales artificial es posible realizar el reconocimiento de las estructuras sintácticas del idioma castellano.
- Que a partir del conocimiento que se tiene de los lenguajes formales se ha podido formalizar el idioma castellano para poder realizar operaciones computacionales.

VII. RECOMENDACIONES

Las recomendaciones para aplicar el método propuesto en la presente investigación son las siguientes:

- Se debe realizar el aprendizaje de la red neuronal artificial con todas las estructuras gramaticales que se quiere reconocer.
- Para el aprendizaje de la red neuronal artificial de una gran cantidad de estructuras se requiere utilizar una computadora de alto desempeño.
- Para implementar en aplicaciones que requieren el uso del lenguaje natural ya sea hablado o escrito no será necesario hacer uso de todas las formas de oración que tiene el idioma castellano.

VIII. REFERENCIAS

Aho,Alfred; Ulman Jeffrey. (2004). *Compiler Principles, techniques & tools.* Boston: Pearson.

Baliri, S. (2014). *Teoria de Lenguajes Formales, Una introduccion para Linguistica.* Barcelona: Universidad Autinoma de Barcelona.

Borja Mario, Torres sally, Lescano sergio. (2009). Recuperacion de datos de manuscritos para crear Bases de Datos, Aplicando redes neuronales artificiales. *Convension Internacional de Informatica*, 1-10.

Breál, M. (1897). *Essai de sémantique: science des significations.* Paris: Librairie Hachette ET.

Bullinaria, J. A. (1997). Modeling Reading, Spelling and Past Tense Learning with Artificial Neural Networks. *Brain and Lenguage*, 236-266.

Chomsky, N. (1965). *Aspect of the Theory of Syntax.* Cambridge: Mit Press.

Cortes, A. (1999). El enfoque biologico del lenguaje . *Investigacion y ciencia, Lenguaje Humano*, 2-7.

Cortez Vasquez, A. M. (2009). Procesamiento de lenguaje natural. *Revista de Ingenieria de Sistemas e Informatica*, 45-54.

Kierkegaard, S. (2006). *o lo uno o lo otro.* España: Rustica.

Sanderson, M. (2000). Retrieving with good sense . *Information Retrieval*, 49-69.

Sosa, E. (1997). Procesamiento del lenguaje natural: revisión del estado actual, bases teóricas y aplicaciones. *Revista Internacional Cientifica y Profesional*, 26-29.

Vallez, M. P. (5 de Mayo de 2007). *El Procesamiento del Lenguaje Natural en la Recuperación de Información Textual y áreas afines*. Obtenido de hipertext.net: www.hipertext.net>

Vemuri, V. (1990). *Artificial Neural Networks : Theorical Concepts.* Los alamitos, CA: Computer Society Press.

IX. ANEXOS

ANEXO 1 Resultado de ejecución del modulo de inicialización de la red neuronal artificial.

PROCESAMIENTO DE LENGUAJE NATURAL

Tipo palabra
Oración
Genera palabras
Asigna tipos de palabras
Genera estructuras
Genera RNA
Aprendizaje
Reconocimiento

Estructura de Red Neuronal Artificial

Estructura	Dato
1	5
2	10
3	15
4	10
5	8
6	1

Red Neuronal Artificial 44

Neurona 1	Capa : 1 Entradas : 10 Sumatoria : 0 Resultado : 0 Umbral : 500 K adaline : 1 peso[1] : 0.3625 peso[2] : 0.2133 peso[3] : 0.5029 peso[4] : 0.6651 peso[5] : 0.0488 peso[6] : 0.7272 peso[7] : 0.1777 peso[8] : 0.1512 peso[9] : 0.2313 peso[10] : 0.5234
Neurona 2	Capa : 1 Entradas : 10 Sumatoria : 0 Resultado : 0 Umbral : 500 K adaline : 1 peso[1] : 0.2791 peso[2] : 0.6562 peso[3] : 0.5503 peso[4] : 0.958 peso[5] : 0.369 peso[6] : 0.6244 peso[7] : 0.6536 peso[8] : 0.4354 peso[9] : 0.9745 peso[10] : 0.9983

Neurona 3	Capa : 1 Entradas : 10 Sumatoria : 0 Resultado : 0 Umbral : 500 K adaline : 1 peso[1] : 0.7416 peso[2] : 0.5141 peso[3] : 0.0121 peso[4] : 0.3843 peso[5] : 0.4387 peso[6] : 0.6442 peso[7] : 0.1113 peso[8] : 0.3094 peso[9] : 0.1084 peso[10] : 0.039
Neurona 4	Capa : 1 Entradas : 10 Sumatoria : 0 Resultado : 0 Umbral : 500 K adaline : 1 peso[1] : 0.4332 peso[2] : 0.471 peso[3] : 0.2523 peso[4] : 0.9362 peso[5] : 0.136 peso[6] : 0.3011 peso[7] : 0.6634 peso[8] : 0.3137 peso[9] : 0.4523 peso[10] : 0.8947
Neurona 5	Capa : 1 Entradas : 10 Sumatoria : 0 Resultado : 0 Umbral : 500 K adaline : 1 peso[1] : 0.8371 peso[2] : 0.7314 peso[3] : 0.5508 peso[4] : 0.3874 peso[5] : 0.6894 peso[6] : 0.9198 peso[7] : 0.0118 peso[8] : 0.3429 peso[9] : 0.3552 peso[10] : 0.9864
Neurona 6	Capa : 1 Entradas : 10 Sumatoria : 0 Resultado : 0 Umbral : 500 K adaline : 1 peso[1] : 0.3411

	peso[2] : 0.0968 peso[3] : 0.5004 peso[4] : 0.3533 peso[5] : 0.4811 peso[6] : 0.9392 peso[7] : 0.9975 peso[8] : 0.5924 peso[9] : 0.2486 peso[10] : 0.1059
Neurona 7	Capa : 1 Entradas : 10 Sumatoria : 0 Resultado : 0 Umbral : 500 K adaline : 1 peso[1] : 0.6315 peso[2] : 0.6818 peso[3] : 0.5769 peso[4] : 0.8838 peso[5] : 0.6179 peso[6] : 0.713 peso[7] : 0.1848 peso[8] : 0.2812 peso[9] : 0.0266 peso[10] : 0.6371
Neurona 8	Capa : 1 Entradas : 10 Sumatoria : 0 Resultado : 0 Umbral : 500 K adaline : 1 peso[1] : 0.1758 peso[2] : 0.8638 peso[3] : 0.3685 peso[4] : 0.7267 peso[5] : 0.2512 peso[6] : 0.0578 peso[7] : 0.6465 peso[8] : 0.263 peso[9] : 0.4008 peso[10] : 0.0016
Neurona 9	Capa : 1 Entradas : 10 Sumatoria : 0 Resultado : 0 Umbral : 500 K adaline : 1 peso[1] : 0.2493 peso[2] : 0.742 peso[3] : 0.0984 peso[4] : 0.7497 peso[5] : 0.0952 peso[6] : 0.5796 peso[7] : 0.6889 peso[8] : 0.0927

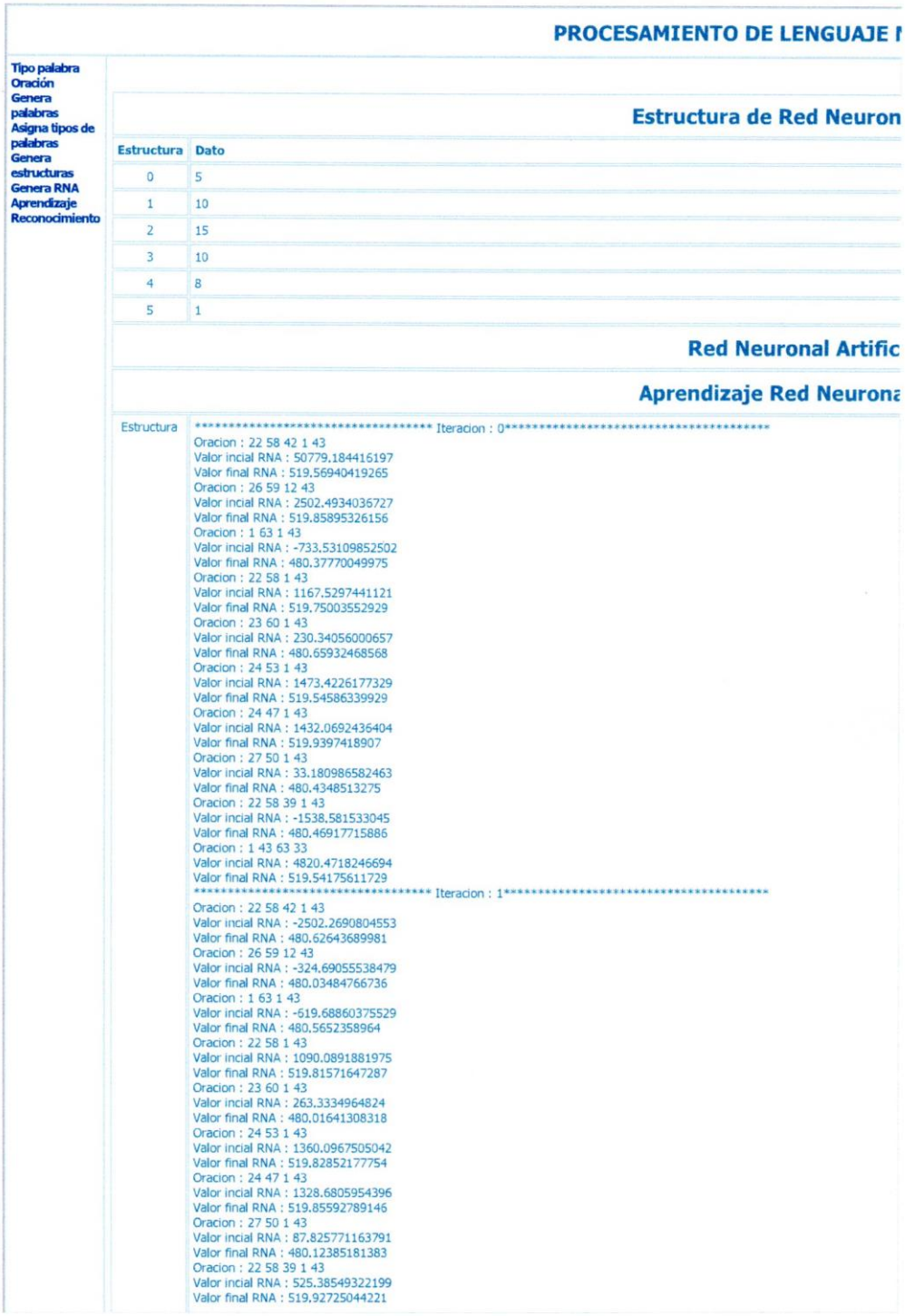
	peso[9] : 0.172 peso[10] : 0.9375
Neurona 10	Capa : 1 Entradas : 10 Sumatoria : 0 Resultado : 0 Umbral : 500 K adaline : 1 peso[1] : 0.1987 peso[2] : 0.8035 peso[3] : 0.6192 peso[4] : 0.7756 peso[5] : 0.6872 peso[6] : 0.2371 peso[7] : 0.4885 peso[8] : 0.872 peso[9] : 0.5183 peso[10] : 0.5152
Neurona 11	Capa : 2 Entradas : 10 Sumatoria : 0 Resultado : 0 Umbral : 500 K adaline : 1 peso[1] : 0.5091 peso[2] : 0.6942 peso[3] : 0.379 peso[4] : 0.8777 peso[5] : 0.4209 peso[6] : 0.6303 peso[7] : 0.9355 peso[8] : 0.0674 peso[9] : 0.8933 peso[10] : 0.3363
Neurona 12	Capa : 2 Entradas : 10 Sumatoria : 0 Resultado : 0 Umbral : 500 K adaline : 1 peso[1] : 0.069 peso[2] : 0.1426 peso[3] : 0.0782 peso[4] : 0.1675 peso[5] : 0.8924 peso[6] : 0.1735 peso[7] : 0.7471 peso[8] : 0.5812 peso[9] : 0.2663 peso[10] : 0.9192
Neurona 13	Capa : 2 Entradas : 10 Sumatoria : 0 Resultado : 0 Umbral : 500

	K adaline : 1 peso[1] : 0.5186 peso[2] : 0.465 peso[3] : 0.7226 peso[4] : 0.1378 peso[5] : 0.2406 peso[6] : 0.4097 peso[7] : 0.3749 peso[8] : 0.7292 peso[9] : 0.2817 peso[10] : 0.8933
Neurona 14	Capa : 2 Entradas : 10 Sumatoria : 0 Resultado : 0 Umbral : 500 K adaline : 1 peso[1] : 0.2444 peso[2] : 0.7909 peso[3] : 0.5875 peso[4] : 0.6234 peso[5] : 0.6685 peso[6] : 0.0084 peso[7] : 0.2536 peso[8] : 0.604 peso[9] : 0.0758 peso[10] : 0.1469
Neurona 15	Capa : 2 Entradas : 10 Sumatoria : 0 Resultado : 0 Umbral : 500 K adaline : 1 peso[1] : 0.9403 peso[2] : 0.1448 peso[3] : 0.2896 peso[4] : 0.0185 peso[5] : 0.3124 peso[6] : 0.1819 peso[7] : 0.1921 peso[8] : 0.0595 peso[9] : 0.7632 peso[10] : 0.4584
Neurona 16	Capa : 2 Entradas : 10 Sumatoria : 0 Resultado : 0 Umbral : 500 K adaline : 1 peso[1] : 0.9787 peso[2] : 0.2817 peso[3] : 0.9235 peso[4] : 0.7012 peso[5] : 0.4195 peso[6] : 0.164

	peso[7] : 0.1109 peso[8] : 0.7945 peso[9] : 0.8932 peso[10] : 0.3927
Neurona 17	Capa : 2 Entradas : 10 Sumatoria : 0 Resultado : 0 Umbral : 500 K adaline : 1 peso[1] : 0.6877 peso[2] : 0.1375 peso[3] : 0.1835 peso[4] : 0.2752 peso[5] : 0.761 peso[6] : 0.852 peso[7] : 0.2836 peso[8] : 0.0146 peso[9] : 0.456 peso[10] : 0.3594
Neurona 18	Capa : 2 Entradas : 10 Sumatoria : 0 Resultado : 0 Umbral : 500 K adaline : 1 peso[1] : 0.1616 peso[2] : 0.3962 peso[3] : 0.5043 peso[4] : 0.4512 peso[5] : 0.4148 peso[6] : 0.8167 peso[7] : 0.6332 peso[8] : 0.6069 peso[9] : 0.8763 peso[10] : 0.3963
Neurona 19	Capa : 2 Entradas : 10 Sumatoria : 0 Resultado : 0 Umbral : 500 K adaline : 1 peso[1] : 0.0653 peso[2] : 0.8549 peso[3] : 0.678 peso[4] : 0.9888 peso[5] : 0.5561 peso[6] : 0.0975 peso[7] : 0.1528 peso[8] : 0.6671 peso[9] : 0.892 peso[10] : 0.0459
Neurona 20	Capa : 2 Entradas : 10 Sumatoria : 0

	Resultado : 0 Umbral : 500 K adaline : 1 peso[1] : 0.0597 peso[2] : 0.5797 peso[3] : 0.1835 peso[4] : 0.2432 peso[5] : 0.8549 peso[6] : 0.9446 peso[7] : 0.0952 peso[8] : 0.1384 peso[9] : 0.9593 peso[10] : 0.5512
Neurona 21	Capa : 2 Entradas : 10 Sumatoria : 0 Resultado : 0 Umbral : 500 K adaline : 1 peso[1] : 0.4979 peso[2] : 0.1208 peso[3] : 0.9475 peso[4] : 0.0021 peso[5] : 0.5721 peso[6] : 0.3622 peso[7] : 0.8189 peso[8] : 0.2052 peso[9] : 0.9692 peso[10] : 0.6951
Neurona 22	Capa : 2 Entradas : 10 Sumatoria : 0 Resultado : 0 Umbral : 500 K adaline : 1 peso[1] : 0.6015 peso[2] : 0.0344 peso[3] : 0.55 peso[4] : 0.2795 peso[5] : 0.0232 peso[6] : 0.1061 peso[7] : 0.377 peso[8] : 0.176 peso[9] : 0.7732 peso[10] : 0.269
Neurona 23	Capa : 2 Entradas : 10 Sumatoria : 0 Resultado : 0 Umbral : 500 K adaline : 1 peso[1] : 0.222 peso[2] : 0.8329 peso[3] : 0.8488 peso[4] : 0.4056

ANEXO 2 Resultado de ejecución del proceso de Aprendizaje de la red neuronal artificial.



PROCESAMIENTO DE LENGUAJE

Tipo palabra
Oración
Genera palabras
Asigna tipos de palabras
Genera estructuras
Genera RNA
Aprendizaje
Reconocimiento

Estructura de Red Neuron

Estructura	Dato
0	5
1	10
2	15
3	10
4	8
5	1

Red Neuronal Artific

Aprendizaje Red Neurona

Estructura

********** Iteracion : 0**********
Oracion : 22 58 42 1 43
Valor incial RNA : 50779.184416197
Valor final RNA : 519.56940419265
Oracion : 26 59 12 43
Valor incial RNA : 2502.4934036727
Valor final RNA : 519.85895326156
Oracion : 1 63 1 43
Valor incial RNA : -733.53109852502
Valor final RNA : 480.37770049975
Oracion : 22 58 1 43
Valor incial RNA : 1167.5297441121
Valor final RNA : 519.75003552929
Oracion : 23 60 1 43
Valor incial RNA : 230.34056000657
Valor final RNA : 480.65932468568
Oracion : 24 53 1 43
Valor incial RNA : 1473.4226177329
Valor final RNA : 519.54586339929
Oracion : 24 47 1 43
Valor incial RNA : 1432.0692436404
Valor final RNA : 519.9397418907
Oracion : 27 50 1 43
Valor incial RNA : 33.180986582463
Valor final RNA : 480.4348513275
Oracion : 22 58 39 1 43
Valor incial RNA : -1538.581533045
Valor final RNA : 480.46917715886
Oracion : 1 43 63 33
Valor incial RNA : 4820.4718246694
Valor final RNA : 519.54175611729
********** Iteracion : 1**********
Oracion : 22 58 42 1 43
Valor incial RNA : -2502.2690804553
Valor final RNA : 480.62643689981
Oracion : 26 59 12 43
Valor incial RNA : -324.69055538479
Valor final RNA : 480.03484766736
Oracion : 1 63 1 43
Valor incial RNA : -619.68860375529
Valor final RNA : 480.5652358964
Oracion : 22 58 1 43
Valor incial RNA : 1090.0891881975
Valor final RNA : 519.81571647287
Oracion : 23 60 1 43
Valor incial RNA : 263.3334964824
Valor final RNA : 480.01641308318
Oracion : 24 53 1 43
Valor incial RNA : 1360.0967505042
Valor final RNA : 519.82852177754
Oracion : 24 47 1 43
Valor incial RNA : 1328.6805954396
Valor final RNA : 519.85592789146
Oracion : 27 50 1 43
Valor incial RNA : 87.825771163791
Valor final RNA : 480.12385181383
Oracion : 22 58 39 1 43
Valor incial RNA : 525.38549322199
Valor final RNA : 519.92725044221

```
Oracion : 1 43 63 33
Valor incial RNA : 2906.7839448068
Valor final RNA : 519.99749298319
********** Iteracion : 2**********
Oracion : 22 58 42 1 43
Valor incial RNA : -1107.4974749021
Valor final RNA : 480.69272438354
Oracion : 26 59 12 43
Valor incial RNA : -933.05272491252
Valor final RNA : 480.29203734437
Oracion : 1 63 1 43
Valor incial RNA : -487.94378940787
Valor final RNA : 480.61009989899
Oracion : 22 58 1 43
Valor incial RNA : 1060.6317642699
Valor final RNA : 519.74937824641
Oracion : 23 60 1 43
Valor incial RNA : 275.40538588942
Valor final RNA : 480.44591121169
Oracion : 24 53 1 43
Valor incial RNA : 1317.7244573019
Valor final RNA : 519.90368995618
Oracion : 24 47 1 43
Valor incial RNA : 1289.3144403796
Valor final RNA : 519.86780775804
Oracion : 27 50 1 43
Valor incial RNA : 108.75139589463
Valor final RNA : 480.28021712766
Oracion : 22 58 39 1 43
Valor incial RNA : 963.53574688844
Valor final RNA : 519.35795679354
Oracion : 1 43 63 33
Valor incial RNA : 2034.0682909978
Valor final RNA : 519.42363579926
********** Iteracion : 3**********
Oracion : 22 58 42 1 43
Valor incial RNA : -495.5480798681
Valor final RNA : 480.04663157548
Oracion : 26 59 12 43
Valor incial RNA : -987.177640462
Valor final RNA : 480.32904321933
Oracion : 1 63 1 43
Valor incial RNA : -390.0405200943
Valor final RNA : 480.40453242104
Oracion : 22 58 1 43
Valor incial RNA : 1045.5915398708
Valor final RNA : 519.735807759
Oracion : 23 60 1 43
Valor incial RNA : 281.80526321102
Valor final RNA : 480.34957594441
Oracion : 24 53 1 43
Valor incial RNA : 1297.1499014255
Valor final RNA : 519.99536031901
Oracion : 24 47 1 43
Valor incial RNA : 1270.4714972468
Valor final RNA : 519.97238960708
Oracion : 27 50 1 43
Valor incial RNA : 118.86872808702
Valor final RNA : 480.30369675793
Oracion : 22 58 39 1 43
Valor incial RNA : 988.0358175324
Valor final RNA : 519.72799088252
Oracion : 1 43 63 33
Valor incial RNA : 1566.0667124877
Valor final RNA : 519.91874701315
********** Iteracion : 4**********
Oracion : 22 58 42 1 43
Valor incial RNA : -167.95163525821
Valor final RNA : 480.513531149
Oracion : 26 59 12 43
Valor incial RNA : -921.92608392985
Valor final RNA : 480.40414648981
Oracion : 1 63 1 43
Valor incial RNA : -319.65948886029
Valor final RNA : 480.38742993594
Oracion : 22 58 1 43
Valor incial RNA : 1037.6937330162
Valor final RNA : 519.4979116158
Oracion : 23 60 1 43
Valor incial RNA : 285.27095275207
Valor final RNA : 480.00856005441
Oracion : 24 53 1 43
Valor incial RNA : 1284.1558725171
Valor final RNA : 519.72163166327
Oracion : 24 47 1 43
Valor incial RNA : 1258.5345478425
Valor final RNA : 519.71215557347
```

```
Oracion : 27 50 1 43
Valor incial RNA : 124.82758320028
Valor final RNA : 480.080131817
Oracion : 22 58 39 1 43
Valor incial RNA : 927.72128536088
Valor final RNA : 519.60333595433
Oracion : 1 43 63 33
Valor incial RNA : 1279.0180725148
Valor final RNA : 519.29355586511
********** Iteracion : 5**********
Oracion : 22 58 42 1 43
Valor incial RNA : 30.37564419458
Valor final RNA : 480.69479309699
Oracion : 26 59 12 43
Valor incial RNA : -847.09880943811
Valor final RNA : 480.53314571193
Oracion : 1 63 1 43
Valor incial RNA : -269.4622318095
Valor final RNA : 480.46596661936
Oracion : 22 58 1 43
Valor incial RNA : 1030.881526267
Valor final RNA : 519.82982125045
Oracion : 23 60 1 43
Valor incial RNA : 288.27777228875
Valor final RNA : 480.2562200265
Oracion : 24 53 1 43
Valor incial RNA : 1274.9465547459
Valor final RNA : 519.53897278182
Oracion : 24 47 1 43
Valor incial RNA : 1249.6768185361
Valor final RNA : 519.52866312489
Oracion : 27 50 1 43
Valor incial RNA : 129.2694784312
Valor final RNA : 480.27854060112
Oracion : 22 58 39 1 43
Valor incial RNA : 864.63006110259
Valor final RNA : 519.42004665245
Oracion : 1 43 63 33
Valor incial RNA : 1091.66923088
Valor final RNA : 519.38957934071
********** Iteracion : 6**********
Oracion : 22 58 42 1 43
Valor incial RNA : 159.07920227787
Valor final RNA : 480.61147840586
Oracion : 26 59 12 43
Valor incial RNA : -785.65540639444
Valor final RNA : 480.36202602539
Oracion : 1 63 1 43
Valor incial RNA : -234.51906098262
Valor final RNA : 480.06785301729
Oracion : 22 58 1 43
Valor incial RNA : 1025.1544893011
Valor final RNA : 519.66192742365
Oracion : 23 60 1 43
Valor incial RNA : 290.39383048498
Valor final RNA : 480.42072915439
Oracion : 24 53 1 43
Valor incial RNA : 1267.2731080743
Valor final RNA : 519.99258016789
Oracion : 24 47 1 43
Valor incial RNA : 1242.8151572944
Valor final RNA : 519.88419391405
Oracion : 27 50 1 43
Valor incial RNA : 133.4977903328
Valor final RNA : 480.46489118035
Oracion : 22 58 39 1 43
Valor incial RNA : 814.80980798155
Valor final RNA : 519.79076607448
Oracion : 1 43 63 33
Valor incial RNA : 965.42663111621
Valor final RNA : 519.78634944355
********** Iteracion : 7**********
Oracion : 22 58 42 1 43
Valor incial RNA : 247.04627459628
Valor final RNA : 480.74005005978
Oracion : 26 59 12 43
Valor incial RNA : -738.58241781786
Valor final RNA : 480.69628995847
Oracion : 1 63 1 43
Valor incial RNA : -207.72885436405
Valor final RNA : 480.09159430968
Oracion : 22 58 1 43
Valor incial RNA : 1020.4425886787
Valor final RNA : 519.52948082451
Oracion : 23 60 1 43
Valor incial RNA : 292.25517957943
Valor final RNA : 480.56321505811
```

```
Oracion : 24 53 1 43
Valor incial RNA : 1260.573378737
Valor final RNA : 519.86685856945
Oracion : 24 47 1 43
Valor incial RNA : 1236.4087646953
Valor final RNA : 519.75962225737
Oracion : 27 50 1 43
Valor incial RNA : 136.72235271166
Valor final RNA : 480.11081240487
Oracion : 22 58 39 1 43
Valor incial RNA : 778.20035346769
Valor final RNA : 519.82060107117
Oracion : 1 43 63 33
Valor incial RNA : 878.69827871138
Valor final RNA : 519.63933207234
********** Iteracion : 8**********
Oracion : 22 58 42 1 43
Valor incial RNA : 306.57640389244
Valor final RNA : 480.32688576537
Oracion : 26 59 12 43
Valor incial RNA : -704.26729964998
Valor final RNA : 480.40890867529
Oracion : 1 63 1 43
Valor incial RNA : -188.59686679431
Valor final RNA : 480.58279391275
Oracion : 22 58 1 43
Valor incial RNA : 1016.5938205174
Valor final RNA : 519.96353280511
Oracion : 23 60 1 43
Valor incial RNA : 294.50105935554
Valor final RNA : 480.04401996004
Oracion : 24 53 1 43
Valor incial RNA : 1253.9793287847
Valor final RNA : 519.7439278528
Oracion : 24 47 1 43
Valor incial RNA : 1230.580917668
Valor final RNA : 519.65072770861
Oracion : 27 50 1 43
Valor incial RNA : 139.65626089849
Valor final RNA : 480.23337711823
Oracion : 22 58 39 1 43
Valor incial RNA : 752.21762526636
Valor final RNA : 519.38345765525
Oracion : 1 43 63 33
Valor incial RNA : 818.56467184991
Valor final RNA : 519.85834597372
********** Iteracion : 9**********
Oracion : 22 58 42 1 43
Valor incial RNA : 347.45178030349
Valor final RNA : 480.04995257854
Oracion : 26 59 12 43
Valor incial RNA : -677.7387033976
Valor final RNA : 480.40501217008
Oracion : 1 63 1 43
Valor incial RNA : -174.70707353314
Valor final RNA : 480.18876310999
Oracion : 22 58 1 43
Valor incial RNA : 1012.3272732216
Valor final RNA : 519.84297859069
Oracion : 23 60 1 43
Valor incial RNA : 296.06097537389
Valor final RNA : 480.16391080156
Oracion : 24 53 1 43
Valor incial RNA : 1248.3321284569
Valor final RNA : 519.64446481719
Oracion : 24 47 1 43
Valor incial RNA : 1225.187111805
Valor final RNA : 519.55239170883
Oracion : 27 50 1 43
Valor incial RNA : 142.38121015597
Valor final RNA : 480.34537839532
Oracion : 22 58 39 1 43
Valor incial RNA : 733.50614966009
Valor final RNA : 519.52609483684
Oracion : 1 43 63 33
Valor incial RNA : 777.28795346014
Valor final RNA : 519.97372591808
********** Iteracion : 10**********
Oracion : 22 58 42 1 43
Valor incial RNA : 376.20728104794
Valor final RNA : 480.11647017644
Oracion : 26 59 12 43
Valor incial RNA : -656.85069122059
Valor final RNA : 480.20807468522
Oracion : 1 63 1 43
Valor incial RNA : -163.7886384954
Valor final RNA : 480.46297298658
```

```
Oracion : 22 58 1 43
Valor incial RNA : 1008.8011844926
Valor final RNA : 519.75009825707
Oracion : 23 60 1 43
Valor incial RNA : 297.55971004683
Valor final RNA : 480.27853596217
Oracion : 24 53 1 43
Valor incial RNA : 1242.9841618938
Valor final RNA : 519.55188100718
Oracion : 24 47 1 43
Valor incial RNA : 1220.0795247023
Valor final RNA : 519.97356174201
Oracion : 27 50 1 43
Valor incial RNA : 145.97864828938
Valor final RNA : 480.50706187283
Oracion : 22 58 39 1 43
Valor incial RNA : 720.2809023686
Valor final RNA : 519.2284109699
Oracion : 1 43 63 33
Valor incial RNA : 748.46146115795
Valor final RNA : 519.51269601325
********** Iteracion : 11**********
Oracion : 22 58 42 1 43
Valor incial RNA : 395.319421858
Valor final RNA : 480.8592774398
Oracion : 26 59 12 43
Valor incial RNA : -640.06475169167
Valor final RNA : 480.49267100423
Oracion : 1 63 1 43
Valor incial RNA : -154.74095886758
Valor final RNA : 480.68457295133
Oracion : 22 58 1 43
Valor incial RNA : 1005.3724564813
Valor final RNA : 519.65927197263
Oracion : 23 60 1 43
Valor incial RNA : 298.99925077736
Valor final RNA : 480.38867022355
Oracion : 24 53 1 43
Valor incial RNA : 1237.8419793785
Valor final RNA : 519.46281697931
Oracion : 24 47 1 43
Valor incial RNA : 1215.1683849365
Valor final RNA : 519.88291898849
Oracion : 27 50 1 43
Valor incial RNA : 148.45703410228
Valor final RNA : 480.05394595296
Oracion : 22 58 39 1 43
Valor incial RNA : 710.96948381819
Valor final RNA : 519.94345362681
Oracion : 1 43 63 33
Valor incial RNA : 728.57350350113
Valor final RNA : 519.60566523986
********** Iteracion : 12**********
Oracion : 22 58 42 1 43
Valor incial RNA : 410.4123991824
Valor final RNA : 480.52490404886
Oracion : 26 59 12 43
Valor incial RNA : -527.29831078372
Valor final RNA : 480.13977597458
Oracion : 1 63 1 43
Valor incial RNA : -147.91466485903
Valor final RNA : 480.14624160448
Oracion : 22 58 1 43
Valor incial RNA : 1001.3477090967
Valor final RNA : 519.54320652044
Oracion : 23 60 1 43
Valor incial RNA : 300.3552376235
Valor final RNA : 480.49215578232
Oracion : 24 53 1 43
Valor incial RNA : 1232.8940877251
Valor final RNA : 519.93531603857
Oracion : 24 47 1 43
Valor incial RNA : 1210.9815535304
Valor final RNA : 519.81027553492
Oracion : 27 50 1 43
Valor incial RNA : 150.85753135
Valor final RNA : 480.1552074963
Oracion : 22 58 39 1 43
Valor incial RNA : 703.96661044483
Valor final RNA : 519.30969048405
Oracion : 1 43 63 33
Valor incial RNA : 714.06663715255
Valor final RNA : 519.06219567586
********** Iteracion : 13**********
Oracion : 22 58 42 1 43
Valor incial RNA : 418.75228472764
Valor final RNA : 480.20842442842
```

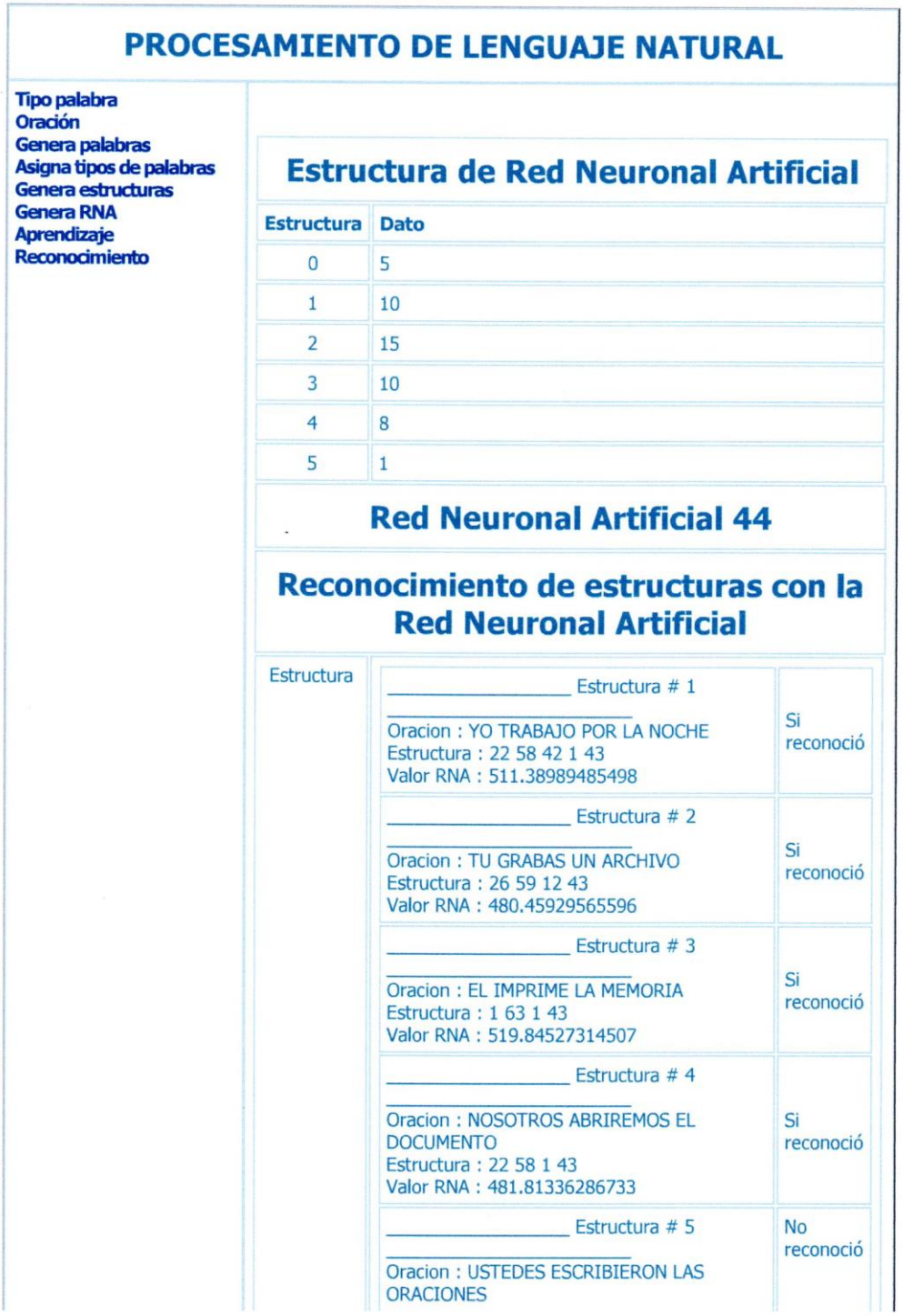
```
Oracion : 26 59 12 43
Valor incial RNA : -616.12211158846
Valor final RNA : 480.31424642574
Oracion : 1 63 1 43
Valor incial RNA : -141.58243459364
Valor final RNA : 480.29701127065
Oracion : 22 58 1 43
Valor incial RNA : 998.06771347163
Valor final RNA : 519.45527213792
Oracion : 23 60 1 43
Valor incial RNA : 301.70791135762
Valor final RNA : 480.59590106533
Oracion : 24 53 1 43
Valor incial RNA : 1228.0536533242
Valor final RNA : 519.84833579601
Oracion : 24 47 1 43
Valor incial RNA : 1206.3553515563
Valor final RNA : 519.72455264613
Oracion : 27 50 1 43
Valor incial RNA : 153.19031670401
Valor final RNA : 480.2533941917
Oracion : 22 58 39 1 43
Valor incial RNA : 698.82162867948
Valor final RNA : 519.97006081265
Oracion : 1 43 63 33
Valor incial RNA : 703.94095318155
Valor final RNA : 519.90071314293
********** Iteracion : 14**********
Oracion : 22 58 42 1 43
Valor incial RNA : 426.45489737901
Valor final RNA : 480.46840751345
Oracion : 26 59 12 43
Valor incial RNA : -605.68266738761
Valor final RNA : 480.47476387733
Oracion : 1 63 1 43
Valor incial RNA : -136.07940320933
Valor final RNA : 480.42302670287
Oracion : 22 58 1 43
Valor incial RNA : 994.83409884524
Valor final RNA : 519.95836065586
Oracion : 23 60 1 43
Valor incial RNA : 303.56053684971
Valor final RNA : 480.09605043274
Oracion : 24 53 1 43
Valor incial RNA : 1222.7375187449
Valor final RNA : 519.74782359676
Oracion : 24 47 1 43
Valor incial RNA : 1201.8112570631
Valor final RNA : 519.64057411739
Oracion : 27 50 1 43
Valor incial RNA : 155.47468119547
Valor final RNA : 480.34928340286
Oracion : 22 58 39 1 43
Valor incial RNA : 694.87117668233
Valor final RNA : 519.60098136911
Oracion : 1 43 63 33
Valor incial RNA : 696.54524880281
Valor final RNA : 519.91024459984
********** Iteracion : 15**********
Oracion : 22 58 42 1 43
Valor incial RNA : 430.97981475289
Valor final RNA : 480.85425198929
Oracion : 26 59 12 43
Valor incial RNA : -596.4367361429
Valor final RNA : 480.60911466737
Oracion : 1 63 1 43
Valor incial RNA : -131.14313278891
Valor final RNA : 480.53243122571
Oracion : 22 58 1 43
Valor incial RNA : 991.63489378932
Valor final RNA : 519.869405291
Oracion : 23 60 1 43
Valor incial RNA : 304.86254893889
Valor final RNA : 480.19959915037
Oracion : 24 53 1 43
Valor incial RNA : 1218.0682730259
Valor final RNA : 519.66417784026
Oracion : 24 47 1 43
Valor incial RNA : 1197.3456503374
Valor final RNA : 519.97182506887
Oracion : 27 50 1 43
Valor incial RNA : 157.72734755085
Valor final RNA : 480.44394868908
Oracion : 22 58 39 1 43
Valor incial RNA : 691.90916965745
Valor final RNA : 519.91899870664
Oracion : 1 43 63 33
```

```
Valor incial RNA : 691.93543535045
Valor final RNA : 519.47031572905
********** Iteracion : 16**********
Oracion : 22 58 42 1 43
Valor incial RNA : 434.47830683458
Valor final RNA : 480.28096621935
Oracion : 26 59 12 43
Valor incial RNA : -588.79436686085
Valor final RNA : 480.0924502948
Oracion : 1 63 1 43
Valor incial RNA : -127.04228527094
Valor final RNA : 480.61572403054
Oracion : 22 58 1 43
Valor incial RNA : 988.44383619171
Valor final RNA : 519.78108149495
Oracion : 23 60 1 43
Valor incial RNA : 306.15170945612
Valor final RNA : 480.30165135294
Oracion : 24 53 1 43
Valor incial RNA : 1213.44389887
Valor final RNA : 519.58211761567
Oracion : 24 47 1 43
Valor incial RNA : 1192.9239790179
Valor final RNA : 519.88900186884
Oracion : 27 50 1 43
Valor incial RNA : 159.95883210056
Valor final RNA : 480.01605187834
Oracion : 22 58 39 1 43
Valor incial RNA : 688.76295608484
Valor final RNA : 519.62018689365
Oracion : 1 43 63 33
Valor incial RNA : 687.29906547871
Valor final RNA : 519.72002584829
********** Iteracion : 17**********
Oracion : 22 58 42 1 43
Valor incial RNA : 437.41891467233
Valor final RNA : 481.15208306299
Oracion : 26 59 12 43
Valor incial RNA : -580.5181682996
Valor final RNA : 480.2099270721
Oracion : 1 63 1 43
Valor incial RNA : -122.74732045861
Valor final RNA : 480.70579565882
Oracion : 22 58 1 43
Valor incial RNA : 985.28374024923
Valor final RNA : 519.69330654817
Oracion : 23 60 1 43
Valor incial RNA : 307.43099577315
Valor final RNA : 480.40322270353
Oracion : 24 53 1 43
Valor incial RNA : 1208.8547922184
Valor final RNA : 519.5002669115
Oracion : 24 47 1 43
Valor incial RNA : 1188.5353203459
Valor final RNA : 519.80638714185
Oracion : 27 50 1 43
Valor incial RNA : 162.17309040932
Valor final RNA : 480.11162529569
Oracion : 22 58 39 1 43
Valor incial RNA : 687.17625469612
Valor final RNA : 519.48265829547
Oracion : 1 43 63 33
Valor incial RNA : 684.02010697712
Valor final RNA : 519.40179540339
********** Iteracion : 18**********
Oracion : 22 58 42 1 43
Valor incial RNA : 439.15771356142
Valor final RNA : 480.53367370748
Oracion : 26 59 12 43
Valor incial RNA : -573.34953551529
Valor final RNA : 480.30580598368
Oracion : 1 63 1 43
Valor incial RNA : -118.62169161816
Valor final RNA : 480.06346476808
Oracion : 22 58 1 43
Valor incial RNA : 980.82795778841
Valor final RNA : 519.55315932311
Oracion : 23 60 1 43
Valor incial RNA : 308.64341515107
Valor final RNA : 480.49786084148
Oracion : 24 53 1 43
Valor incial RNA : 1204.3106314173
Valor final RNA : 519.99982530149
Oracion : 24 47 1 43
Valor incial RNA : 1185.2471027588
Valor final RNA : 519.75519803654
Oracion : 27 50 1 43
```

```
Valor incial RNA : 164.39386752686
Valor final RNA : 480.20774080076
Oracion : 22 58 39 1 43
Valor incial RNA : 685.39410681067
Valor final RNA : 519.32451461219
Oracion : 1 43 63 33
Valor incial RNA : 681.43912963712
Valor final RNA : 519.93218332617
********** Iteracion : 19**********
Oracion : 22 58 42 1 43
Valor incial RNA : 440.94251796373
Valor final RNA : 480.49535530939
Oracion : 26 59 12 43
Valor incial RNA : -566.27226080874
Valor final RNA : 480.39990138374
Oracion : 1 63 1 43
Valor incial RNA : -114.62934876669
Valor final RNA : 480.14858335096
Oracion : 22 58 1 43
Valor incial RNA : 977.74620589452
Valor final RNA : 519.46761859808
Oracion : 23 60 1 43
Valor incial RNA : 309.891387348
Valor final RNA : 480.59687164731
Oracion : 24 53 1 43
Valor incial RNA : 1199.8341272603
Valor final RNA : 519.92366440616
Oracion : 24 47 1 43
Valor incial RNA : 1180.9720298808
Valor final RNA : 519.67457933618
Oracion : 27 50 1 43
Valor incial RNA : 166.55114454523
Valor final RNA : 480.30108082645
Oracion : 22 58 39 1 43
Valor incial RNA : 683.24885135018
Valor final RNA : 519.80858863454
Oracion : 1 43 63 33
Valor incial RNA : 680.0148770873
Valor final RNA : 519.80248592866
********** Iteracion : 20**********
Oracion : 22 58 42 1 43
Valor incial RNA : 442.00957874566
Valor final RNA : 480.12701068787
Oracion : 26 59 12 43
Valor incial RNA : -559.26260680816
Valor final RNA : 480.4892106613
Oracion : 1 63 1 43
Valor incial RNA : -110.71234498491
Valor final RNA : 480.23245941746
Oracion : 22 58 1 43
Valor incial RNA : 974.68288210688
Valor final RNA : 519.96691707581
Oracion : 23 60 1 43
Valor incial RNA : 311.71396179965
Valor final RNA : 480.07469077744
Oracion : 24 53 1 43
Valor incial RNA : 1194.2175340266
Valor final RNA : 519.80869275185
Oracion : 24 47 1 43
Valor incial RNA : 1176.6755495047
Valor final RNA : 519.59371221368
Oracion : 27 50 1 43
Valor incial RNA : 168.70039499725
Valor final RNA : 480.39367657673
Oracion : 22 58 39 1 43
Valor incial RNA : 681.41508193122
Valor final RNA : 519.63777029045
Oracion : 1 43 63 33
Valor incial RNA : 678.22108651529
Valor final RNA : 519.63222880345
********** Iteracion : 21**********
Oracion : 22 58 42 1 43
Valor incial RNA : 442.78503082306
Valor final RNA : 480.37966904658
Oracion : 26 59 12 43
Valor incial RNA : -552.0488850263
Valor final RNA : 480.58218926396
Oracion : 1 63 1 43
Valor incial RNA : -106.87997924872
Valor final RNA : 480.31324850611
Oracion : 22 58 1 43
Valor incial RNA : 971.62906046659
Valor final RNA : 519.87876590329
Oracion : 23 60 1 43
Valor incial RNA : 312.94586391211
Valor final RNA : 480.17665149661
Oracion : 24 53 1 43
```

```
Valor incial RNA : 1189.7934237189
Valor final RNA : 519.72684102703
Oracion : 24 47 1 43
Valor incial RNA : 1172.4325954799
Valor final RNA : 519.51372008405
Oracion : 27 50 1 43
Valor incial RNA : 170.83859425151
Valor final RNA : 480.48617867042
Oracion : 22 58 39 1 43
Valor incial RNA : 679.99375088544
Valor final RNA : 519.5110077718
Oracion : 1 43 63 33
Valor incial RNA : 676.57124933423
Valor final RNA : 519.47723310307
********** Iteracion : 22**********
Oracion : 22 58 42 1 43
Valor incial RNA : 443.52076555457
Valor final RNA : 480.61909343838
Oracion : 26 59 12 43
Valor incial RNA : -545.08104393866
Valor final RNA : 480.0583724458
Oracion : 1 63 1 43
Valor incial RNA : -103.72761720094
Valor final RNA : 480.37262027892
Oracion : 22 58 1 43
Valor incial RNA : 968.57609966961
Valor final RNA : 519.78993918566
Oracion : 23 60 1 43
Valor incial RNA : 314.16818212714
Valor final RNA : 480.27812166949
Oracion : 24 53 1 43
Valor incial RNA : 1185.3993913211
Valor final RNA : 519.64508047825
Oracion : 24 47 1 43
Valor incial RNA : 1168.2178083803
Valor final RNA : 519.43380928851
Oracion : 27 50 1 43
Valor incial RNA : 172.96195929754
Valor final RNA : 480.03402868163
Oracion : 22 58 39 1 43
Valor incial RNA : 677.96990431721
Valor final RNA : 519.31899355875
Oracion : 1 43 63 33
Valor incial RNA : 674.61434923986
Valor final RNA : 519.28835570996
********** Iteracion : 23**********
Oracion : 22 58 42 1 43
Valor incial RNA : 444.41646355299
Valor final RNA : 480.21355592003
Oracion : 26 59 12 43
Valor incial RNA : -539.21118701397
Valor final RNA : 480.12784057075
Oracion : 1 63 1 43
Valor incial RNA : -100.06789767839
Valor final RNA : 480.44825960198
Oracion : 22 58 1 43
Valor incial RNA : 965.55270308448
Valor final RNA : 519.70253655544
Oracion : 23 60 1 43
Valor incial RNA : 315.38597744825
Valor final RNA : 480.37900552106
Oracion : 24 53 1 43
Valor incial RNA : 1181.0251800936
Valor final RNA : 519.56411050987
Oracion : 24 47 1 43
Valor incial RNA : 1164.0224442344
Valor final RNA : 519.35470161628
Oracion : 27 50 1 43
Valor incial RNA : 175.07624580487
Valor final RNA : 480.12845586196
Oracion : 22 58 39 1 43
Valor incial RNA : 676.89087178625
Valor final RNA : 519.90829614192
Oracion : 1 43 63 33
Valor incial RNA : 673.83783792708
Valor final RNA : 519.2292532119
********** Iteracion : 24**********
Oracion : 22 58 42 1 43
Valor incial RNA : 445.29917786324
Valor final RNA : 480.51513206713
Oracion : 26 59 12 43
Valor incial RNA : -532.57110143257
Valor final RNA : 480.21308629761
Oracion : 1 63 1 43
Valor incial RNA : -96.399452272413
Valor final RNA : 480.52475640824
Oracion : 22 58 1 43
```

ANEXO 3 Resultado de ejecución del proceso de reconocimiento de estructuras con la red neuronal artificial.



PROCESAMIENTO DE LENGUAJE NATURAL

Tipo palabra
Oración
Genera palabras
Asigna tipos de palabras
Genera estructuras
Genera RNA
Aprendizaje
Reconocimiento

Estructura de Red Neuronal Artificial

Estructura	Dato
0	5
1	10
2	15
3	10
4	8
5	1

Red Neuronal Artificial 44

Reconocimiento de estructuras con la Red Neuronal Artificial

Estructura		
	_____ Estructura # 1 _____ Oracion : YO TRABAJO POR LA NOCHE Estructura : 22 58 42 1 43 Valor RNA : 511.38989485498	Si reconoció
	_____ Estructura # 2 _____ Oracion : TU GRABAS UN ARCHIVO Estructura : 26 59 12 43 Valor RNA : 480.45929565596	Si reconoció
	_____ Estructura # 3 _____ Oracion : EL IMPRIME LA MEMORIA Estructura : 1 63 1 43 Valor RNA : 519.84527314507	Si reconoció
	_____ Estructura # 4 _____ Oracion : NOSOTROS ABRIREMOS EL DOCUMENTO Estructura : 22 58 1 43 Valor RNA : 481.81336286733	Si reconoció
	_____ Estructura # 5 _____ Oracion : USTEDES ESCRIBIERON LAS ORACIONES	No reconoció

Estructura : 23 60 1 43 Valor RNA : 471.43127869894	
_____ Estructura # 6 _____ Oracion : ELLOS PROBARAN LA FUNCION Estructura : 24 53 1 43 Valor RNA : 495.51572038856	Si reconoció
_____ Estructura # 7 _____ Oracion : ELLAS SACARON LOS VIDEOS Estructura : 24 47 1 43 Valor RNA : 518.49340429391	Si reconoció
_____ Estructura # 8 _____ Oracion : ELLA ENCONTRO LA PAGINA Estructura : 27 50 1 43 Valor RNA : 498.8359937414	Si reconoció
_____ Estructura # 9 _____ Oracion : NOSOTROS JUGAMOS EN LA COMPUTADORA Estructura : 22 58 39 1 43 Valor RNA : 507.74435715335	Si reconoció
_____ Estructura # 10 _____ Oracion : LA IMPRESORA ES MIA Estructura : 1 43 63 33 Valor RNA : 494.42549530255	Si reconoció
_____ Estructura # 11 _____ Oracion : EL TECLADO ES MIO Estructura : 1 43 63 33 Valor RNA : 494.42549530255	Si reconoció
_____ Estructura # 12 _____ Oracion : ESA CAMARA ES MIA Estructura : 17 43 63 33 Valor RNA : 450.85979610342	No reconoció
_____ Estructura # 13 _____ Oracion : ESE MOUSE ES TUYO Estructura : 18 43 63 32 Valor RNA : 430.40162290075	No reconoció
_____ Estructura # 14 _____ Oracion : EL CELULAR ES SUYO Estructura : 1 43 63 33 Valor RNA : 494.42549530255	Si reconoció
_____ Estructura # 15 _____	Si reconoció

Oracion : LA COMPUTADORA ES SUYA Estructura : 1 43 63 33 Valor RNA : 494.42549530255	
_____ Estructura # 16 _____ Oracion : ESTE ES MI DOCUMENTO Estructura : 18 63 10 43 Valor RNA : 484.49333085088	Si reconoció
_____ Estructura # 17 _____ Oracion : ESTA ES MI PAGINA Estructura : 17 63 10 43 Valor RNA : 487.21618705082	Si reconoció
_____ Estructura # 18 _____ Oracion : ESTOS SON TUS ARCHIVOS Estructura : 16 60 10 43 Valor RNA : 501.42788520344	Si reconoció
_____ Estructura # 19 _____ Oracion : ESTAS SON MIS CLASES Estructura : 15 60 10 43 Valor RNA : 504.15074140338	Si reconoció
_____ Estructura # 20 _____ Oracion : ESA CARPETA ES MIA Estructura : 17 43 63 33 Valor RNA : 450.85979610342	No reconoció
_____ Estructura # 21 _____ Oracion : ESE ARCHIVO SE CERRO Estructura : 18 43 50 Valor RNA : -152.92585122689	No reconoció
_____ Estructura # 22 _____ Oracion : AQUEL QUE TRABAJA TANTO ES MI HERMANO Estructura : 18 21 63 63 10 43 Valor RNA : 20379.158829999	No reconoció
_____ Estructura # 23 _____ Oracion : AQUELLA ES MI SOLICITUD Estructura : 17 63 10 43 Valor RNA : 487.21618705082	Si reconoció
_____ Estructura # 24 _____ Oracion : AQUELLOS JUEGOS SON ADICTIVOS Estructura : 16 43 60 2 Valor RNA : -99.857712482789	No reconoció

_____ Estructura # 25 _____ Oracion : AQUELLAS PAGINAS SON DE MI TESIS Estructura : 15 43 60 10 43 Valor RNA : 769.38517725225	No reconoció
_____ Estructura # 26 _____ Oracion : ALGUIEN APAGO LA COMPUTADORA Estructura : 20 61 1 43 Valor RNA : 475.77023331455	No reconoció
_____ Estructura # 27 _____ Oracion : NADIE TOMO NOTA EN LA CLASE Estructura : 20 50 43 39 1 43 Valor RNA : 19661.034805881	No reconoció
_____ Estructura # 28 _____ Oracion : QUE FUNCIONE BIEN POR FAVOR Estructura : 21 42 Valor RNA : -218.02376753626	No reconoció
_____ Estructura # 29 _____ Oracion : CUAL DE SUS AMIGOS LLEGO PRIMERO Estructura : 43 50 Valor RNA : -308.56351580885	No reconoció
_____ Estructura # 30 _____ Oracion : QUIENES VINIERON CON ELLAS Estructura : 47 24 Valor RNA : -219.88497701882	No reconoció

Procesamiento de lenguaje natural