Comportamiento Adaptable de Chatbots Dependiente del Contexto

Juan Manuel Rodríguez[1], Hernán Merlino[1,2], Enrique Fernández[1,2]
1. Cátedra de Sistemas de Programación no Convencional de Robots. Facultad de Ingeniería. Universidad de Buenos Aires
2. Laboratorio de Investigación y Desarrollo en Arquitecturas Complejas (LIDAC). Grupo de Investigación en Sistemas de Información (GISI). Universidad Nacional de Lanús. Argentina
hmerlino@gmail.com

***Resumen*—El objeto de este trabajo es el estudio de los sistemas llamados chatbots, sus limitaciones actuales y sus posibilidades de mejora. En particular aquellas basadas en la interpretación pragmática del discurso o contexto. Se proponen una serie de algoritmos tendientes a mejorar un chatbot existente: ALICE utilizando el algoritmo RAP (Resolution of Anaphora Procedure) y una red neuronal artificial.**

***Palabras Clave*—Resolution of Anaphora Procedure (RAP), AIML, chatbot, ALICE, Perceptron, SOM, Kohone, PLN.**

I. INTRODUCCIÓN

El tema de este trabajo de investigación son los programas (*software*) llamados comúnmente chatbots. Estos son algoritmos de computadora, que utilizan procesamiento de lenguaje natural (*NLP: Natural Language Processing*) en un sistema de preguntas y respuestas (*QA systems: question-answering systems.*) Estos sistemas han sido definidos también como sistemas expertos que usan razonamiento basado en casos (*CBR: case base reasoning*) [19].

El ideal perseguido detrás de estos sistemas es que puedan "comportarse" de forma indistinguible de un ser humano, donde en este caso "comportarse" significa que puedan dar respuestas satisfactorias a un interlocutor humano tal y como lo haría una persona durante un dialogo. Hoy en día los chatbots existentes están lejos de lograr este cometido aunque existen sistemas que han dado respuestas suficientemente satisfactorias como para engañar a un interlocutor humano al menos durante unos instantes, ejemplos de estos diálogos pueden verse en [11]; en este trabajo se analizarán en detalles los sistemas ALICE y ELIZA.

El interés detrás del ideal de funcionamiento de un chatbot va desde lo filosófico, hasta lo pragmático, el matemático británico Alan Turing sostuvo en [18] que un chatbot capaz de pasar cierta prueba, hoy conocida como Test de Turing era un fuerte indicio de que las maquinas podían pensar. Por otro lado el creador de AIML sostiene en [19] que un sistema capaz de comunicarse mediante el dialogo con un ser humano podría convertirse en un nuevo tipo de interfaz de usuario, en donde el usuario dejaría de utilizar comandos, ventanas, iconos, etc. para simplemente indicarle al sistema lo que quiere hacer de forma meramente coloquial.

A pesar de las obvias limitaciones de los chatbots actuales muchos son utilizados comercialmente de forma más o menos exitosa, son utilizados como agentes automáticos en videojuegos, asistentes de mesa de ayuda virtual o "amigos virtuales" en redes de mensajería (se pueden ver ejemplos concretos en la sección II.B.4)

Finalmente cabe destacar que el problema del correcto funcionamiento de los chatbots se enmarca dentro de la problemática del procesamiento del lenguaje natural, la cual es una de las áreas más complejas dentro de los llamados sistemas inteligentes.

A. Objetivos del trabajo

El objetivo del presente trabajo es buscar una forma de mejorar la calidad de las respuestas y del dialogo en general que puede generar un chatbot; para ello se tomará una tecnología (en el terminó más amplio posible, entiéndase: lenguaje de programación, conjunto de datos, metodología de trabajo y/o comportamiento general de un algoritmo) utilizada en un chatbot existente y se mejorará su comportamiento en relación con la interpretación pragmática del discurso (o dialogo en este caso). Más adelante, en la sección III.A se detallarán algunos de los problemas encontrados en los chatbots, como ser la dificultad que tienen estos sistemas para dar una respuesta satisfactoria cuando esta depende de elementos presentes en el "contexto", como ser la resolución anafórica de referentes.

II. ESTADO DE LA CUESTIÓN

El presente capítulo describe los algoritmos y sistemas relevantes sobre el tema de estudio al momento de escribir esta investigación así como los avances científicos en el área. La primera sección muestra el estado de los chatbots actuales en general (II.A), la segunda sección presenta la utilización de sistemas inteligentes en relación con el procesamiento de lenguaje natural (II.B), en la tercer sección (II.C) se hace una introducción a la problemática actual, la cual será desarrollada en el próximo capítulo y por último en la cuarta sección (II.D) del presente capítulo se detallan las tecnologías que serán utilizadas en la resolución del problema.

A. Chatbots

Los chatbots son programas, *software*, (entiéndase un conjuntos de algoritmos), que utilizan procesamiento de lenguaje natural *(NLP: Natural Language Processing)* en un sistema de preguntas y respuestas *(QA systems: question-answering systems)* [2]. Estos sistemas han sido definidos también como sistemas expertos que usan razonamiento basado en casos (*CBR: case base reasoning)* [19]. La finalidad de dichos sistemas es simular un dialogo inteligente con interlocutor humano, ya sea mediante mensajes de texto a través de una consola o bien mediante la voz.

1) Orígenes del planteo

En 1950 el matemático británico Alan Turing en una publicación científica propuso una forma de responder al interrogante: "¿Pueden pensar las maquinas?" [18]. Su método consistía en la implementación de un juego al que llamó: *The imitation game* (el juego de la imitación) en el cual participan dos personas y una maquina. Una de las personas actúa como juez o jurado y debe discernir quien de los restantes es la persona. Por supuesto que tanto la otra persona como la maquina no están en la misma habitación que el jurado, y este solo puede comunicarse con ambas mediante preguntas escritas, las cuales son respondidas en igual forma. Turing sostuvo que si la maquina podía engañar al jurado por una cantidad de tiempo a definir, entonces se podría considerar que dicha maquina piensa. A esta prueba se la llama comúnmente: *Test de Turing*

Al día de hoy no ha sido creado un sistema capaz de pasar de forma satisfactoria dicha prueba.

La premisa de Turing de que una máquina capaz de pasar dicha prueba es una máquina que puede pensar fue rebatida por Jhon Searle [16] con el argumento de la habitación china. También Roger Penrose [14] ha rebatido mediante otros argumentos el punto de vista de Turing.

2) Primera aproximación real: ELIZA

El primer chatbot que logró responder a interrogantes hechos a través de una consola de texto y confundir a una persona al punto de no saber esta si en verdad estaba hablando con una maquina o no fue ELIZA. Por supuesto que dicha confusión se daba solo en las primeras líneas del dialogo y con determinadas frases.

ELIZA fue diseñado en 1966 por Joseph Weizenbaum. El chatbot parodia a un psicólogo en su manera de responder ya que en general sus respuestas son reformulaciones de las frases de entrada del usuario [20]

De forma esquemática, el algoritmo detrás de ELIZA busca palabras claves en una frase de entrada de un usuario y luego utilizando el resto de la oración hace una reformulación de la sentencia original, esta forma de responder solo es coherente en el ámbito del psicoanálisis, entre las partes: paciente y doctor. Por ejemplo: ante la frase de entrada: "It seems that you hate me", ELIZA detectaría las palabras claves: "you" y "me". Aplicaría luego un *template* que descompondría la frase original en cuatro partes:

(1) It seems that (2) you (3) hate (4) me.

Finalmente realizaría la reformulación de la frase original, una respuesta posible sería: "What makes you think **I hate you**" [20]. ELIZA tiene además otros mecanismos para armar respuestas, pero el descripto es el principal.

3) Concurso Loebner (Loebner Prize)

El Premio Loebner de Inteligencia Artificial (*The Loebner Prize for artificial intelligence*) es la primera instancia formal de un *Test de Turing*. En 1990 Hugh Loebner acordó con el Centro de Estudios del Comportamiento de Cambridge (The Cambridge Center for Behavioral Studies) diseñar un certamen designado para implementar el *Test de Turing*. El Dr. Loebner ofreció un premio de 100 000 dólares y una medalla de oro para la primera computadora cuyas respuestas fueran indistinguibles de respuestas humanas. Todos los años en dicho certamen se otorga un premio de 2.000 dólares y una medalla de bronce a la computadora que se acerca más a un ser humano en su forma de responder. El ganador del certamen anual es el mejor en relación con los otros concursantes de ese mismo año y no en un sentido absoluto [8]

Los primeros años los premios fueron ganados por chatbots que implementaban el funcionamiento básico de ELIZA. Sin embargo los años subsiguientes aparecieron otros tipos de chatbots más sofisticados que empezaron a ocupar los primeros puestos. Una tecnología que apareció para el desarrollo de chatbots más complejos fue AIML y en particular una implementación de chatbot con dicha tecnología: ALICE salió vencedor del certamen en tres oportunidades. [11] [1]

Alguno de los chatbots que han participado en el certamen se detallan debajo:

TABLA I. DETALLES PC THERAPIST

Nombre	**PC THERAPIST**
Descripción	Primer programa en ganar el Premio Loebner. Este chatbot fue escrito en 1986 y está basado en ELIZA
Creador	Joseph Weintraub

TABLA II. DETALLES JULIA

Nombre	**Julia**
Descripción	Chatbot basado en Eliza, participó en el concurso Loebner del año 1994, salió cuarta de cinco *chatbots*.
Disponible	http://www.lazytd.com/lti/julia/

TABLA III. DETALLES BRIAN

Nombre	**Brian**
Descripción	Este *chatbot* está escrito en C++, extiende la idea general del "terapista" que utiliza ELIZA. Ganó el tercer lugar en el concurso Loebner 1998
Disponible	http://www.strout.net/info/science/ai/brian/

TABLA IV. DETALLES ALICE

Nombre	**ALICE**
Descripción	ALICE está desarrollado sobre AIML y fue ganador del premio Loebner en tres oportunidades.
Disponible	http://www.pandorabots.com/pandora/talk?botid=f5d922d97e345aa1

TABLA V. DETALLES EUGENE GOOSTMAN

Nombre	**Eugene Goostman**
Descripción	Segundo en el concurso Loebener de 2005.
Disponible	http://www.mangoost.com/bot/

TABLA VI. DETALLES JABBERWACKY

Nombre	**Jabberwacky**
Descripción	Jabberwacky fue ganador del premio Loebner en dos oportunidades.
Disponible	http://www.jabberwacky.com/

TABLA VII. DETALLES ULTRA HAL

Nombre	Ultra Hal
Descripción	Este chatbot ganó el primer lugar en el concurso Loebner del año 2007.
Disponible	http://www.zabaware.com/webhal/index.html

Las transcripciones de las conversaciones entre los jurados y los chatbots están todas disponibles de forma *on-line*. Además algunos de los chatbots están disponibles en internet para que cualquier persona pueda interactuar con ellos, sin embargo solo un par están disponibles para que cualquiera pueda observar su funcionamiento, de los últimos ganadores del certamen, ALICE es uno de estos.

4) Implementación y utilización comercial de chatbots

Si bien los chatbots aún tienen demasiados problemas para que sean adoptados como una tecnología solida hay casos puntuales en donde su utilización puede considerarse exitosa. Como ejemplos se pueden nombrar ciertos chatbots creados para el mensajero instantáneo de Microsoft (Messenger o MSN). Una empresa llamada IMI desarrolló un intérprete de AIML en C# para que sea posible crear en este lenguaje chatbots para Microsoft Messenger y ya se dispone de una lista de chatbots existentes para MSN, incluyendo algunos como Robin o Wilma creados para un propósito especifico distinto del mero interés académico.

Robin es un chatbot creado por el Ministerio de Sanidad y Consumo de España. Su función es informar a los jóvenes a través de Messenger sobre enfermedades de transmisión sexual y consumo de alcohol.

Microsoft creó su propio chatbot, Encarta Instant Answer, para promocionar su enciclopedia encarta, sin embargo con la finalización de dicho producto por parte de Microsoft el chabot fue dado de baja.

Algunos sistemas informáticos como Remedy, utilizan chatbots para que los usuarios puedan evacuar consultas comunes de manera rápida, estos chatbots simulan un operario de *Help Desk*.

B. Procesamiento de Lenguaje Natural

La presente sección describe la utilización de sistemas inteligentes en relación con el procesamiento de lenguaje natural como se anticipó.

1) Pragmática

Si bien una cadena de palabras puede tener una o varias interpretaciones semánticas, estas interpretaciones pueden ser incompletas si no se añade la información dependiente del contexto sobre la situación actual de cada interpretación posible.

La pragmática o pragmalingüística es un subcampo de la lingüística que se interesa por el modo en que el contexto influye en la interpretación del significado. El contexto debe entenderse como situación, ya que puede incluir cualquier aspecto extralingüístico: situación comunicativa, conocimiento compartido por los hablantes, relaciones interpersonales, etc. La pragmática toma en consideración los factores extralingüísticos que condicionan el uso del lenguaje, esto es, todos aquellos factores a los que no se hace referencia en un estudio puramente formal.

Como en una conversación vía *chat*, dos entidades que se comunican no comparten más que sus palabras, todo el contexto queda limitado a eso, a lo que se dijo anteriormente. A pesar de la reducción notable de lo que habría que evaluar para deducir el contexto, el problema sigue siendo de una complejidad enorme.

2) Resolución de referentes

Según [15]: "La necesidad más obvia de información pragmática es en la resolución del significado de referentes, que son frases que hacen referencia directamente a la situación actual. Por ejemplo en, en la frase «yo estoy en Boston hoy», la interpretación de los referentes «yo» y «hoy» depende de quién declare la frase.[...] El oyente que percibe un acto de habla debería también percibir qué hablante es y usar esta información para resolver el referente." Es decir que el oyente debería de poder discernir quien es "yo" y cuando es "hoy".

La resolución por referencia es la interpretación de un pronombre o una frase nominativa definitiva que hace referencia a un objeto en el mundo (En lingüística, mencionar algo que ya ha sido introducido se llama referencia anafórica. Referenciar algo que todavía tiene que ser introducido se llama referencia catafórica). La resolución se basa en el conocimiento del mundo en las partes previas al discurso. Consideremos el pasaje:

«Juan paró al camarero. Él pidió un sándwich de jamón.»

Para entender que «él» en la segunda oración se refiere a Juan, necesitamos haber entendido que en la primera oración menciona dos personas y que Juan está ejerciendo el papel de cliente y por ello es capaz de pedir mientras que el camarero no lo es. Usualmente, la resolución de referencias es un problema en donde se debe seleccionar una referencia de una lista de candidatas." [15]

Han sido diseñados una gran cantidad de algoritmos de resolución por referencia, dos de los más conocidos son, el algoritmo *pronoun references* [6] y *Pronominal Anaphora Resolution* [7]

C. Problemática actual

En la actualidad según las publicaciones anuales de los resultados del certamen: *Loebner Prize* (http://www.loebner.net/Prizef/loebner-prize.html) no hay diálogos entre chatbots y personas que logren pasar de forma satisfactoria el t*est de Turing*, tampoco de que dichas conversaciones se asemejen a una conversación humana o incluso coherente. Hay varios problemas a superar, pero sin duda uno de los más evidentes es la interpretación pragmática de la conversación. Los chatbots no pueden seguir el hilo de una conversación por mucho tiempo (dando respuestas que satisfagan a su interlocutor) ya que no logran resolver las referencias anafóricas ya sean de tipo correferencial, de sentido o elípticas:

Sin embargo logran dar respuestas suficientemente satisfactorias a preguntas o frases armadas, en donde no hay ninguna referencia anafórica, como por ejemplo: "¿Qué edad tienes tú?" o "Mi nombre es Juan". Si bien en el primer ejemplo hay un único referente: "tú" este referencia directamente al chatbot y no es un referente que necesita ser buscado en el dialogo, como sería el caso de "¿Qué edad tiene él?". Este tipo de frases son mencionadas a lo largo de este trabajo como frases: "independientes del contexto" en oposición a las otras que son mencionadas como frases: "dependientes del contexto"

La mitad de la problemática reside en poder distinguir cuando una frase es "independiente del contexto" y cuando es "dependiente del contexto", la otra mitad es como transformar

las frases "dependientes del contexto" en frases "independientes del contexto".

D. Tecnología a utilizar

Se presentan en esta sección las tres tecnologías principales que serán combinadas para lograr mejorar la interpretación pragmática del dialogo en un chatbot. La primera es: redes neuronales artificiales; más específicamente la red *back propagation* o de retropropagación, que se explica maqs adelante en el punto y la red SOM (Kohnonen) o auto organizada que se explica en parrafos siguientes; la segunda tecnología es el algoritmo *Pronominal Anaphora Resolution*, por último se describirá el lenguaje AIML y su mecánica que es la base sobre la cual está construido el chatbot: ALICE que es el punto de partida del presente trabajo.

1) Redes neuronales artificiales

Las redes neuronales artificiales (RNA) imitan la estructura física del sistema nervioso, con la intención de construir sistemas de procesamiento de la información paralelos, distribuidos y adaptativos, que puedan presentar un cierto comportamiento «inteligente»" [12]

Para ampliar la definición anterior se debe añadir que están compuestas por pequeñas unidades llamadas neuronas artificiales, las cuales pueden recibir información de entrada, en general valores numéricos dentro de un rango (o valores de tensión para implementaciones de *hardware*), la cual pueden modificar y luego devolver como salida. Estas unidades pueden conectarse entre sí uniendo las entradas de unas a las salidas de otras formando verdaderas redes, incluso estas interconexiones, llamadas sinapsis, pueden ser ponderadas, por un valor llamado peso que modifica el valor de salida de una neurona antes de que ingrese en otra.

Las neuronas que reciben los datos iniciales conforman lo que se conoce como: "capa de entrada" mientras que las que devuelven el último valor sin que este lo recoja ninguna otra neurona conforman la "capa de salida". Entre ambas capas puede haber otras más, llamadas capas ocultas.

Estos conceptos pueden ser definidos de una manera mucho más rigurosa desde un punto de vista matemático. A continuación se da la definición de [13]:

Una red neuronal artificial es un grafo dirigido, con las siguientes propiedades:

1) A cada nodo *i* se asocia una variable de estado x_i.
2) A cada conexión (*i,j*) de los nodos *i* y *j* se asocia un peso $w_{ij} \in \Re$
3) A cada nodo *i* se asocia un umbral θ_i.
4) Para cada nodo *i* se define una función $f_i(x_j, w_{ij}, \theta_i)$, que depende de los pesos de sus conexiones, del umbral y de los estados de los nodos *j* a él conectados. Esta función proporciona el nuevo estado del nodo.

Donde los nodos son las neuronas y las conexiones son las sinapsis en la terminología más común. Y según la definición anterior:

Neurona de entrada: neurona sin sinapsis entrantes.

Neurona de salida: neurona sin sinapsis salientes

Neurona oculta: toda neurona que no es de entrada ni de salida.

a) Perceptrón multicapa

Un perceptrón multicapa (*MLP: Multi-Layer Perceptron*) es un perceptrón simple con capas ocultas añadidas. Esta arquitectura suele entrenarse mediante el algoritmo denominado retropropagación de errores o *Back Propagation* o bien haciendo uso de alguna de sus variantes o derivados, motivo por el que en muchas ocasiones el conjunto arquitectura MLP + Aprendizaje con *Back Propagation* suele denominarse red de retropropagación o simplemente *Back Propagation* [12].

El perceptrón simple es un modelo neuronal unidireccional, compuesto por dos capas de neuronas, una de entrada y otra de salida (Figura 1). La operación de una red de este tipo, con **n** neuronas de entrada y **m** de salida, se puede expresar como:

$$y_i(t) = f\left(\sum_{j=1}^{n} w_{ij} x_j - \theta_i\right), \quad \forall i, 1 \le i \le m \tag{1}$$

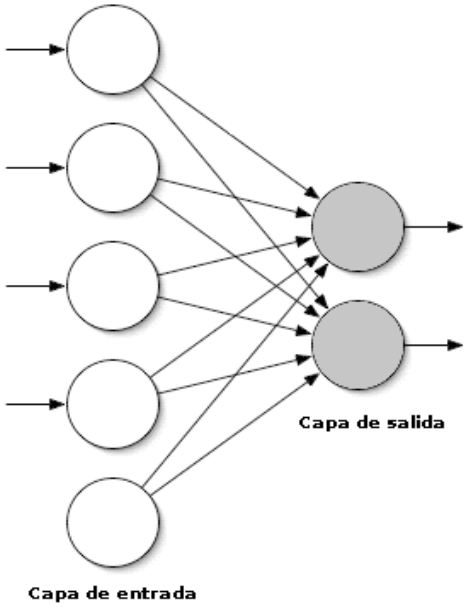

Fig. 1. Perceptrón simple

Las neuronas de entrada no realizan ningún computo, únicamente envían la información a las neuronas de salida. La función de activación de las neuronas de la capa de salida podría ser por ejemplo la función escalón o bien la función *sigmoidal* que son dos de las más utilizadas.

Para poder expresar la función matemática de la forma de operatoria de un *MLP* con una capa oculta y neuronas de salida lineal, como el que se muestra en la Figura 2 denominaremos x_i a las entradas de la red, y_i a las salidas de la capa oculta y z_k a las de la capa final (y globales de la red); t_k serán las salida objetivo (*target*). Por otro lado, w_{ij} son los pesos de la capa oculta y θ_j sus umbrales, w'_{kj} los pesos de la capa de salida y θ'_k *sus umbrales*. Entonces tenemos una expresión como la siguiente:

$$Z_k = \sum_j w'_{kj}\, y_i - \theta'_i = \sum_j w'_{kj}\; f\left(\sum_i w_{ji} x_i - \theta_j\right) - \theta'_i \tag{2}$$

Siendo $f()$ de tipo *sigmoidal* (Figura 3) como por ejemplo:

$$f(x) = 1 / (1 + e^{-x}) \tag{3}$$

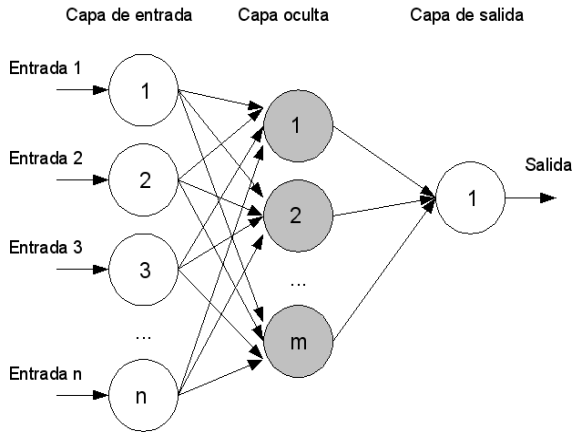

Fig. 2. MLP

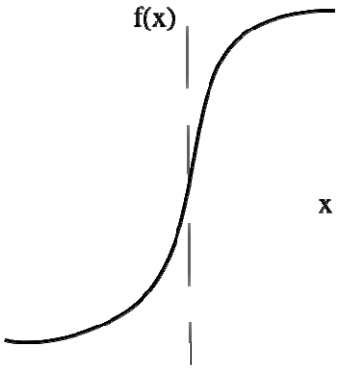

Fig. 3. función sigmoidal

b) Perceptrón multicapa como aproximador universal de funciones

En 1989 Funahashi [3] demostró que un *MLP* de una única capa oculta constituía un aproximador universal de funciones (aunque hubo otras demostraciones más o menos simultáneas.)

Teorema de [3]: Sea *f(x)* una función no constante, acotada y monótona creciente. Sea K un subconjunto compacto (acotado y cerrado) de $\Re^n$. Sea un número real $\varepsilon \in \Re$, (error) y sea un entero $k \in \mathbf{Z}$, tal que $k \geq 3$, que fijamos (cantidad de capas). En estas condiciones se tiene que:

Cualquier *mapping* $g{:}x \in K \rightarrow (g_1(x), g_2(x),...,g_m(x)) \in \Re^m$, con $g_i(x)$ sumables en K, puede ser aproximado en el sentido de la topología L_2 en K por el *mapping* entrada-salida representado por una red neuronal unidireccional (*MLP*) de *k* capas(*k-2* ocultas), con *f(x)* como función de transferencia de las neuronas ocultas, y funciones líneales para las capas de entrada y de salida. En otras palabras:

$\forall \ \varepsilon > 0$, $\exists$ un *MLP* de las características anteriores que implementa el *mapping:*

$$\mathbf{g'}{:}\mathbf{x} \in \mathbf{K} \rightarrow (g'_1(\mathbf{x}), g'_2(\mathbf{x}),...,g'_m(\mathbf{x})) \in \Re^m \tag{4}$$

de manera que:

$$d_{L2(k)} = (g,g') = \left(\sum_{i=1}^{m} \int \| g_i(x_1,....,x_n) - g'_i(x_1,....,x_n)\| \, dx\right)^{1/2} < \varepsilon \tag{5}$$

En resumen un *MLP* de una única capa oculta (podría ser cualquier cantidad de estas) puede aproximar hasta el nivel deseado cualquier función continua en un intervalo, por lo tanto, las redes neuronales multicapa unidireccionales son aproximadores universales de funciones.

c) Back Propagation

El concepto de propagar hacia atrás los errores, durante el proceso de actualización de los pesos sinápticos da lugar al nombre del algoritmo.

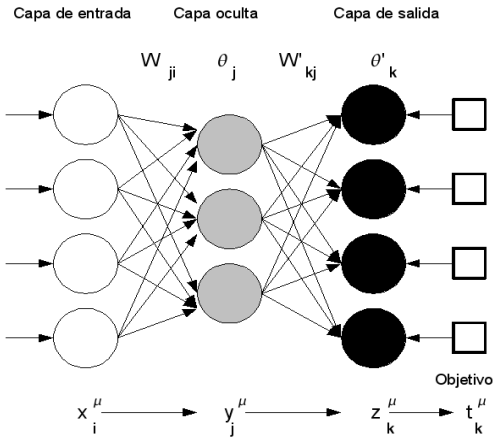

Fig. 4. MLP y Back Propagation

En primer lugar se calcula la expresión denominada: "señal de error" la cual es proporcional al error de salida actual de la red.

Con este valor se calcula la actualización de los pesos sinápticos de la capa de salida. A continuación se propagan hacia atrás los errores (señal de error) a través de las sinapsis de la capa oculta; con esta se calcula la actualización de los pesos sinápticos de las capas ocultas. Puede realizarse con cualquier cantidad de capas ocultas.

Sea un *MLP* de tres capas cuya arquitectura se presenta en la Figura 4. Y sea un patrón de entrada:

$$X^{\mu} = (\mu=1,..,p):$$

La operación global de esta arquitectura se expresa del siguiente modo:

$$Z^{\mu}_k = \sum w'_{kj}\, y^{\mu}_j - \theta'_k = \sum_j w'_{kj} \quad f\Big(\sum_i w_{ji}\, x^{\mu}_i - \theta_j\Big) - \theta'_k \tag{6}$$

las funciones de activación de las neuronas ocultas $f(h)$ son de tipo sigmoidal, con *h* el potencial postsináptico o local. La función "coste" de la que se parte es el error cuadrático medio:

$$E\,(w_{ji}, \theta_j, w'_{kj}, \theta'_j) = (1/2)\sum_{\mu}\sum_{k}[t^{\mu}_k - f(\sum_j w'_{kj} - y^{\mu}_j - \theta'_k)]^2 \tag{7}$$

La minimización se lleva a cabo mediante el descenso por el gradiente, como en este ejemplo hay una capa oculta, se tendrá un gradiente respecto de los pesos de la capa de salida y otro respecto de los de la capa oculta. Las expresiones son las siguientes:

$$\delta w'_{kj} = -\varepsilon \frac{\partial E}{\partial w'_{kj}} \quad \delta w'_{ji} = -\varepsilon \frac{\partial E}{\partial w'_{ji}}$$

(8)

Las expresiones de actualización de los pesos se obtienen sólo con derivar teniendo en cuenta las dependencias funcionales y aplicando adecuadamente la regla de la cadena:

$$\delta w'_{kj} = \varepsilon \sum_{\mu} \Delta'^{\mu}_{k} y^{\mu}_{j} \text{ , con } \Delta'^{\mu}_{k} = [\, t^{\mu}_{k} - f(v'^{\mu}_{k})] \frac{\partial f(v'^{\mu}_{k})}{\partial v'^{\mu}_{k}}$$

(9)

$$\delta w'_{ji} = \varepsilon \sum_{\mu} \Delta^{\mu}_{j} x^{\mu}_{i} \text{ , con } \Delta^{\mu}_{j} = \left(\sum_{k} \Delta'^{\mu}_{k} w'_{kj}\right) \frac{\partial f(v^{\mu}_{j})}{\partial v^{\mu}_{j}}$$

(10)

La actualización de los umbrales (*bias*) se realiza haciendo uso de estas mismas expresiones, considerando que el umbral es un caso particular de peso sináptico cuya entrada es una constante igual a -1. [12]

De forma esquemática sus pasos son los siguientes:

1. Inicializar los pesos y los umbrales iniciales de cada neurona. Hay varias posibilidades de inicialización siendo las más comunes las que introducen valores aleatorios pequeños.
2. Para cada patrón del conjunto de los datos de entrenamiento:
    a. Obtener la respuesta de la red ante ese patrón. Esta parte se consigue propagando la entrada hacia adelante, ya que este tipo de red *(MLP*) es *feedforward*. Las salidas de una capa sirven como entrada a las neuronas de la capa siguiente, procesándolas de acuerdo a la regla de propagación y la función de activación correspondientes.
    b. Calcular los errores asociados según las ecuaciones 2.9 y 2.10
    c. Calcular los incrementos parciales (sumandos de las sumatorias). Estos incrementos dependen de los errores calculados en 2.b
3. Calcular el incremento total, para todos los patrones, de los pesos y los umbrales según las expresiones en las ecuaciones 2.9 y 2.10
4. Actualizar pesos y umbrales
5. Calcular el error actual y volver al paso 2 si no es satisfactorio.

d) Redes SOM, Kohonen

Se hará una breve introducción a este tipo de red ya que finalmente no fue utilizada en la implementación de la solución. Aunque sí fueron utilizadas durante las fases de prueba.

En 1982 T. Kohonen presentó un modelo de red neuronal con capacidad para formar mapas de características de manera similar a como ocurre en el cerebro. Este modelo tiene dos variantes denominadas *LCQ* (*Learning vector quantization*) y TPM (*Topologu Preserving Map*) o SOM (*Self Organizing Map*). Ambas se basan en el principio de formación de mapas topológicos para establecer características comunes entre informaciones (vectores) de entrada de la red, aunque difieren en las dimensiones de estos, siendo de una sola dimensión en el caso de *LVQ* y bidimensional e incluso tridimensional en la red *TPM*. [García Martínez,Servente,Pasquini 2003]

La arquitectura de dicha red puede ser descripta como de dos capas: con N neuronas de entrada y M se salida (Figura 5)

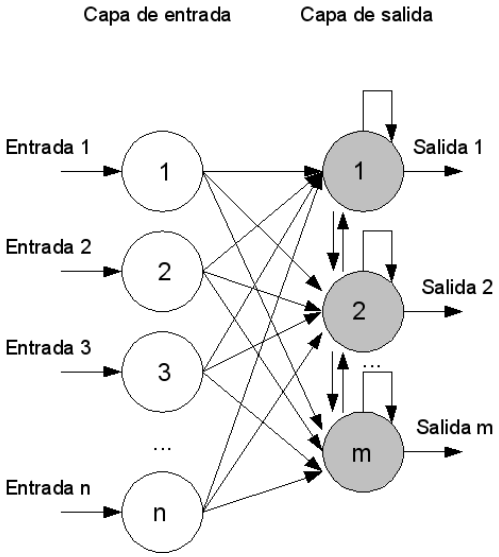

Fig. 5. Estructura de una red de Kohonen

Cada una de las N neuronas de entrada se conecta a las M de salida a través de conexiones hacia adelante (*feedforward*). Entre las neuronas de la capa de salida, puede decirse que existen conexiones laterales de inhibición (peso negativo) implícitas, pues aunque no estén conectadas, cada una de estas neuronas va a tener cierta influencia sobre sus vecinas.

La influencia que cada neurona ejerce sobre las demás es función de las distancia entre ellas, siendo muy pequeñas cuando están muy alejadas. Es frecuente que dicha influencia tenga la forma de que se muestra en la Figura 6.

La versión del modelo denominado *TPM* trata de establecer una correspondencia entre los datos de entrada y un espacio bidimensional de salida, creando mapas topológicos de dos dimensiones, de tal forma que ante datos de entrada con características comunes se deben activar neuronas situadas en zonas próximas de la capa de salida:

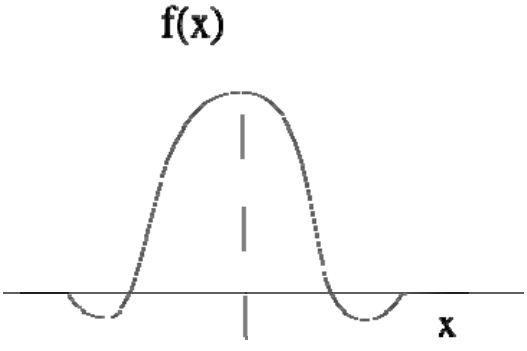

Fig. 6. función de interacción lateral entre neuronas

e) Funcionamiento de una red de Kohonen

Dado un vector de entrada E_k, tal que $E_k = (e_1, e_2, e_3, ...e_n)$, para una red como la descripta anteriormente con N neuronas de entrada y M de salida; la salida general de una neurona de salida j será:

$$s_j(t+1) = f\left(\sum_{i=1}^{N} w_{ij}\, e_i + \sum_{p=1}^{M} Int_{pj}\, s_p(t)\right) \quad (11)$$

Donde Int_{pj} es una función como la de la Figura 6 que representa la influencia lateral de la neurona p sobre la neurona j. La función de activación de las neuronas de salida $f(x)$ será del tipo continuo, líneal o sigmoidal, ya que esta red trabaja con valores reales.

La red revoluciona hasta alcanzar un estado estable en el que solo hay una neurona activada, la ganadora. La formulación matemática del funcionamiento de esta red puede simplificarse así [5]:

$$s_j = \begin{cases} 1 & MIN \,||\, E_k - W_j \,|| = MIN \left(\sum_{i=1}^{N} (e_i - w_{ij})^2\right)^{1/2} \\ 0 & resto \end{cases} \quad (12)$$

Donde $||\, E_k - W_j \,||$ es una medida de la diferencia entre el vector de entrada y el vector de pesos de las conexiones que llegan a la neurona j desde la entrada. Es en estos pesos donde se registran los datos almacenados por la red durante el aprendizaje. Durante el funcionamiento, lo que se pretende es encontrar el dato aprendido más parecido al de entrada para averiguar qué neurona se activará y en qué zona del espacio bidimensional de salida se encuentra. [4]

2) Algoritmo Resolution of Anaphora Procedure

RAP (*Resolution of Anaphora Procedure*) es un algoritmo para identificar los sintagmas nominales (*NP: Noun Phrase*) antecedentes de pronombres en tercera persona y anáforas léxicas (reflexivas y reciprocas). El algoritmo se aplica a las propiedades sintácticas de una oración generadas con algún *parser* de procesamiento de lenguaje natural.

El sintagma es un tipo de constituyente sintáctico (palabra o secuencia de palabras que funcionan en conjunto como una unidad dentro de la estructura jerárquica de la oración) formado por un grupo de palabras que forman otros sub-constituyentes, al menos uno de los cuales es un núcleo sintáctico. Las propiedades combinatorias de un sintagma se derivan de las propiedades de su núcleo sintáctico. Por su parte el núcleo sintáctico es la palabra que da sus características básicas a un sintagma y es por tanto el constituyente más importante o de mayor jerarquía que se encuentra en su interior. RAP contiene los siguientes componentes principales [7]:

- Un filtro sintáctico *intraoracional* para descartar la dependencia anafórica de un pronombre en una *NP* por razones sintácticas
- Un filtro morfológico para descartar la dependencia anafórica de un pronombre en una *NP* por falta de coincidencias con persona, número o género.
- Un procedimiento de identificación de un pleonasmo
- Un algoritmo de vinculación de anáforas para la identificación de posibles antecedentes asignados de una anáfora léxica dentro de la misma sentencia.
- Un procedimiento para la asignación de valores a varios parámetros de relevancia (función gramatical, paralelismo entre funciones gramaticales, frecuencia de mención, proximidad) de un *NP*. Este procedimiento emplea una jerarquía de funciones gramaticales según la cual se asignan pesos más relevantes a:
  (i) sujeto sobre no sujeto
  (ii) los objetos directos más otros complementos,
  (iii) los argumentos de un verbo por sobre complementos y objetos de frases preposicionales (*PP: Prepositional Phrase*) que actúan como complemento del verbo (si la frase preposicional está en el predicado, es complemento del verbo)
  (iv) los núcleos sustantivos por sobre los complementos de los núcleos sustantivos
- Un procedimiento para identificar NPs vinculados anafóricamente como clases equivalentes para las cuales el valor global prominente es la suma de los valores de sus elementos
- Un procedimiento de decisión para seleccionar el elemento preferido de una lista de antecedente candidatos para un pronombre.

3) AIML

AIML (*Artificial Intelligence Mark-up Language*) permite a las personas ingresar conocimiento en chatbots basado en la tecnología de software libre ALICE.

AIML fue desarrollado por la comunidad Alicebot de software libre y el Dr. Richard Wallace durante los años 1995-2000. Fue adaptado de una gramática que no era XML aunque también se llamaba AIML y formó las bases para el primer Alicebot, A.L.I.C.E., (*Artificial Linguistic Internet Computer Entity*).

AIML describe una clase de objetos de datos llamados objetos AIML y (parcialmente) el comportamiento de los programas de computadora que procesan dichos objetos. Los objetos AIML están formados por unidades llamadas: "topic" y "category", que contienen datos analizados los cuales componen los elementos AIML que encapsulan conocimiento de la forma estímulo-respuesta. [19]

La unidad básica de conocimiento en AIML es una "category" como se mencionó. Cada *category* se compone de una pregunta de entrada, una respuesta de salida y un contexto opcional. La pregunta, o estímulo, se llama: "pattern". La respuesta se llama: "template". Los dos tipos de contexto opcionales se denominan: "that" y "topic". El lenguaje de los patrones (*patterns*) en AIML es simple, consiste solamente en palabras, espacios y los símbolos *wildcard* "_" (guión bajo) y "*" (asterisco). Que representan tanto una palabra, como una serie de palabras o bien nada, la única diferencia entre ellos es la prioridad. Las palabras pueden contener letras y números pero no otros caracteres. El lenguaje de los patrones es *case insensitive*. Las palabras están separadas por un espacio único.

Uno de los aspectos más importantes de AIML es que implementa recursividad con el operador: "srai". Como menciona el Dr. Wallace en [19] este operador tiene muchos usos:

- Reducción simbólica: Reduce las formas gramaticales complejas a otras más simples.
- Dividir sentencias largas en dos o más subdivisiones y combinar las respuestas de cada una.
- Sinónimos: Muchas entradas diferentes se pueden vincular a una única respuesta.

- Ortografía: Se puede utilizar para vincular una entrada mal escrita hacia la entrada correcta.
- Detección de palabras clave en cualquier lugar de la entrada.
- Condicionales: Ciertas formas de ramificación pueden ser implementadas con "srai"
- Cualquier combinación de las anteriores.

AIML funciona sobre XML, por lo que la base de datos de conocimientos de un chatbot implementado con AIML no es otra cosa que uno o más archivos XML con varios *tags* de tipo "category" como el que se muestra a continuación:

```
<category>
    <pattern>ARE YOU RELATED TO ALICE *</pattern>
        <template>
                Alice <person/> has been an influence on me.
        </template>
</category>
```

Si un chatbot AIML tuviese esta *category* implementada y un usuario ingresase una frase que comenzase con: "Are you related to Alice", el chabot respondería con la frase: "Alice <person/> has been an influence on me." Donde el *tag* "<person/>" sería cambiado por el contenido de la frase original que coincidió con el *wildacard; por* ejemplo si la frase de entrada hubiese sido: "Are you related to Alice Cooper", la respuesta sería: "Alice Cooper has been an influence on me."

a) Manejo del contexto en AIML

Como se mostró en el punto anterior AIML funciona como una gran tabla de dos columnas: patrones de coincidencia (*patterns*) y salidas (*templates*); en donde para una frase de entrada dada se busca el mejor patrón de coincidencia y se devuelve la respuesta asociada a dicho patrón. Esta salida podría ser, si está al operador "srai", una nueva frase de entrada, en cuyo caso se devolvería la nueva salida asociada.

Sin embargo AIML implementa tres mecanismos para manejar el contexto, o la interpretación pragmática estos son:

- El tag "that": que contiene una frase que podría haber sido dicha por el chatbot en la línea anterior del dialogo.
- Las variables de sesión, variables que pueden ser completadas dinámicamente durante una conversación para ser utilizadas luego.
- El tag "topic": que contiene un tema sobre el cual se podría estar hablando en un dialogo entre el chatbot y el usuario

El uso del tag *that* será ilustrado con el siguiente dialogo de ejemplo:

chatbot: Do you like movies?
usuario: yes
chatbot: I like movies too.

Queda claro que la frase de entrada: "yes" no basta para dar la respuesta satisfactoria del ejemplo, por lo que AIML no buscará simplemente un pattern que coincida con "yes" sino que además buscará que el tag "that" coincida con "Do you like movies?". De esta forma se pueden tener muchas categories con patterns "yes" útiles para diferentes circunstancias.

Se muestra a continuación un ejemplo del uso de las variables de sesión, las cuales se *setean* en algunas *categories*, tomese una *category* como la siguiente:

```
<category>
        <pattern>MY NAME IS MIKE</pattern>
        <template>
        Hi <set name="name">Mike</set>, I know someone else named Mike too.
        </template>
</category>
```

Si el usuario ingresó una frase como: "My name is Mike", además de responder con el contenido del *tag* "template" el chatbot *seteará* una variable de sesión llamada: "name" con el nombre: "Mike". Más tarde podrá usar el contenido de esta variable para armar una respuesta a otra entrada diferente.

El *tag topic* funciona en parte como una variable más de sesión, para alguna *category* puede *setearse* la variable de sesión: "topic" para indicar que se está hablando de un tema en particular, por ejemplo:

```
<category>
                <pattern>yes</pattern>
                <that>Do you like movies?<that>
                <template>
                 I like movies too.
                </template>
                <think>
                        <set name="topic">movies</set>
                </think>
</category>
```

Luego a medida que avanza la conversación, el intérprete AIML podrá verificar además de los *tags pattern* y *that* un tercero, que está por sobre *category* y es el *tag topic*. Al igual que el *tag that* no es un *tag* obligatorio pero si existe las *categories* dentro de él tendrán mayor prioridad que las otras.

b) ALICE

ALICE es el chatbot más exitoso de todos los implementados con la tecnología AIML. En 2003 contaba con una base de datos de más 40 000 *categories*. Fue presentado en el certamen *The Loebner Prize for artificial intelligence* (mencionado en el punto: II.A.3) y ganó en tres oportunidades: 2000, 2001 y 2004 [11]. Y en otras llegó a ser finalista.

ALICE corre sobre "Program D" un intérprete de AIML creado en 2001 y antes de eso corría sobre otro intérprete llamado "Program B"; al momento de escribir este informe ALICE se encuentra de forma *online* para que cualquier persona pueda interactuar con dicho chatbot en la siguiente dirección: "http://alice.pandorabots.com/" y la totalidad de los archivos AIML que componen su base de datos, también llamada "cerebro de ALICE" se encuentra disponible de forma libre y gratuita.

III. Definición del problema

En este capítulo se describe el problema abordado en esta investigación comenzando con una descripción de la problemática actual (sección III.A) la cual se reducirá a ciertos casos puntuales y bien definidos que comprenderán el alcance de este trabajo (sección III.B); finalmente en la sección III.C se muestran ejemplos de casos reales obtenidos de las transcripciones de conversaciones entre personas y chatbots del concurso Loebner.

A. Problemática actual

Uno de los problemas que enfrentan los chatbots actualmente es la interpretación pragmática del discurso. La mayoría de los chatbots que, solo pueden ser analizados a

través de las respuestas que dan, muestran un comportamiento que parece obviar toda información, o buena parte de esta, del dialogo excepto la frase de entrada inmediata. Aún restringiendo la situación comunicativa al conocimiento compartido por los hablantes en el dialogo puntual que se está llevando a cabo. En el caso de AIML, una de las tecnologías actualmente más solidas para la construcción de este tipo de sistemas, el chatbot responde a una frase de entrada según con que patrón de su base de datos de conocimiento coincidió. En algunos casos toma en consideración una variable llamada "topic", ver punto 2.4.3.1 (Manejo del contexto en AIML), para determinar que el patrón de coincidencia pertenezca al mismo tópico de la conversación y en otros verifica también la respuesta dada anteriormente (uso del tag "that".) Sin embargo como muestra el análisis hecho a partir de las transcripciones de las conversaciones del certamen Loebner, estos mecanismos no son suficientes para una completa evaluación del contexto.

Esta falta de "lectura" del contexto, de no considerar en cada nueva respuesta a todo el dialogo como un *input* y solo tomar como tal a la última frase, lleva al chatbot a cometer un *error* al generar la respuesta (se considerará error a toda respuesta a una frase de entrada del usuario que no satisface las expectativas de este de igual forma en que lo haría la respuesta de un interlocutor humano en condiciones similares como en el caso de un *test* de Turing) Dichos errores pueden ser provocados por alguno de los problemas que se listan a continuación:

Resolución de referentes: La resolución del significado de referentes o de anáforas se da cuando se quiere interpretar un pronombre que aparece en la oración y hace referencia a un objeto previamente introducido en el discurso. Ya que por ejemplo el chabot, podría contar en su base de datos con una respuesta satisfactoria para la frase: "Who is Richard Wallace?", sin embargo si un usuario ingresa: "Who is him?" y "him" hiciese referencia a "Richard Wallace" el chatbot no daría una respuesta satisfactoria.

Ambigüedad Léxica: Este problema se da cuando una palabra o frase tiene más de un significado posible y este queda revelado solo en la evaluación del contexto de la conversación, por ejemplo la frase: "The batter hit the ball" parece tener una interpretación no ambigua en la cual un jugador de baseball golpea la bola, pero obtendremos una interpretación diferente si la frase previa es: "the mad scientist unleashed a tidal wave of cake mix towards the ballroom." (En la primer oración "batter" parecería significar: "bateador", "hit": "golpeó" y "ball": pelota, sin embargo al anteponer la segunda oración "batter" significa "masa", "hit": "alcanzó" y "ball": "baile")

Elipsis: La elipsis en lingüística se refiere a ciertas construcciones sintácticas en las que no aparece alguna palabra que se refiera a una entidad lógica necesaria para el sentido de la frase. Por ejemplo en la frase: "Yo corría por la vereda y ellos por la calle". Se omite el verbo "correr" anterior a "por la calle". Incluso hay casos aún más complejos dentro de la problemática de los chatbots como el de la sentencia: "I couldn't open the door" en donde la oración está completa (no falta en principio ninguna palabra) y está correctamente formada pero no contiene mucha información. En el caso de AIML una sentencia de entrada como esa podría coincidir con un patrón con *wildcards* y devolver una respuesta genérica relacionada con abrir puertas. Sin embargo si la sentencia anterior fuese: "I got home and", se estaba dando más información y si pudieran ser combinadas ambas sentencias para buscar un patrón, se lograría una respuesta más acorde. Suponiendo, por supuesto, que haya un patrón de coincidencia en AIML para una frase similar a: "I couldn't open [my] home door", por ejemplo. La solución parcial a este problema en AIML es el uso de la variable "topic", esta debería de *setease* en "home" cuando el usuario ingrese la primer frase, y luego tendría que existir un patrón de coincidencia para: "I couldn't open the door" dentro de dicho *topic*.

Referencias inexistentes: Este caso es el inverso al primero, el usuario hace referencia a algo o alguien que nunca fue introducido en el discurso y el chatbot responde como si "supiese" de quien o de que se está hablando. Por ejemplo, si un usuario comienza una conversación con "Do you like her?", el chatbot probablemente dará una respuesta del tipo: "yes, I like." que podría pasar desapercibida en la mayoría de los casos pero no en este escenario; No respondería como lo haría un ser humano quien inmediatamente preguntaría por la referencia insatisfecha del pronombre: *her*. Ejemplo: "Who is her?" o "Who are you talking about?"

Respuestas a interrogaciones cerradas: En muchos casos el chatbot hace preguntas al usuario y luego de que este responde, el chatbot no logra generar una nueva respuesta acorde a las expectativas de su interlocutor. Muchos de esos casos se dan con interrogaciones cerradas, preguntas que pueden ser respondidas con sí o con no ("yes" o "no".) Por ejemplo, si el chatbot emite como "respuesta" la siguiente pregunta cerrada: "Do you like movies?" y el usuario simplemente ingresa como respuesta: "yes" o bien "no", el chatbot buscará a continuación (es el caso de AIML) un patrón que coincida con "yes" o con "no", según sea el caso y terminará dando una respuesta genérica sin relación con el tema de conversación: "movies" en este caso. En cambio si el usuario hubiese ingresado una respuesta como: "yes, I like movies", el chatbot buscaría un patrón de coincidencia con dicha sentencia y encontraría una respuesta más acorde. AIML intenta solucionar este problema de forma más o menos satisfactoria con el *tag* "that", sin embargo la base de datos de ALICE no tiene entradas para todos los casos de preguntas cerradas posibles dentro del conjunto de preguntas que puede formular el propio chatbot.

B. Alcance propuesto

Los problemas mencionados en el punto anterior, no conforman la totalidad de problemas devenidos de la falta de interpretación pragmática pero son los más evidentes. Si se compara la forma en la cual trabaja el cerebro humano al dialogar, ya sea interpretando oraciones que recibe de su interlocutor o generando las respuestas, con la forma en la cual funcionan los chatbots quedarían en evidencia problemas más profundos y difíciles de resolver, pero eso escapa al objetivo de esta investigación.

El trabajo se centrará en tres problemas básicos, de los mencionados arriba, relacionados con la interpretación pragmática: resolución de referentes, referencias inexistentes y respuestas a interrogantes cerrados.

Para el caso de la resolución de referentes se utilizará el algoritmo: *RAP* (*Resolution of Anaphora Procedure*) de Lappin-Leass. Dicho algoritmo ha dado buenos resultados, 74 % de tasa de éxito en la resolución de anáforas ínter-sentencias y 89 % de tasa de éxito en la resolución de anáforas intra-sentencias [7]

Para el caso de resolución de referencias inexistentes, se tomarán los casos en donde el algoritmo anterior no haya devuelto referencias y se formulará una pregunta que indague sobre la referencia insatisfecha.

Finalmente se implementará un algoritmo sencillo de reemplazos, para convertir las preguntas en afirmaciones para los casos de respuestas monosilábicas.

Con las soluciones planteadas, que se desarrollarán en el punto siguiente (4), se espera tener un éxito moderado, sabiendo que no solucionarán en su totalidad dichos problemas, se pretende reducir la tasa de respuestas erróneas cada vez que el chatbot se encuentre con alguno de los problemas mencionados.

La implementación de las soluciones propuestas plantea un nuevo problema, que es la conservación de los casos favorables, es decir aquellos casos "pregunta-respuesta" que funcionan correctamente en la actualidad y no necesitan ningún tipo de reformulación ni cambio. La manera de solucionar este nuevo problema será clasificar los casos en favorables y no favorables mediante una red.

C. Ejemplos de casos reales

Los siguientes ejemplos fueron obtenidos de las transcripciones del certamen Loebner 2009 [11]:

- http://loebner.net/Prizef/2009_Contest/loebner-prize-2LL009.html, 2006:
- http://loebner.net/Prizef/2006_Contest/loebner-prize-2006.html y 2004:
- http://loebner.net/Prizef/2004_Contest/loebner-prize-2004.html

Ejemplo 1)

A continuación se muestra un problema de ambigüedad léxica y el error en la respuesta del chatbot presentado por: David Levy, llamado: "Do-Much-More" quien resultó vencedor en el certamen del año 2009. (Extracto de la conversación el Juez 2)

chatbot: I don't suppose cars will burn coal, but I wouldn't now for sure. Everything about cars is confidential.

human: Are you experiments confidential?

chatbot: Yes I am experiments, but don't worry. Experiments don't bite!

En este ejemplo se muestra como el chatbot "interpretó" de una forma diferente la sentencia de la persona. Probablemente porque internamente la frase de entrada del usuario hizo coincidencia con un patrón con wildcards (como sería en AIML o ELIZA) "Are you * ", donde la mayoría de las veces "Are you" significa "Eres tu…" y no "Son tus..."

Ejemplo 2)

A continuación se muestra un problema de elipsis y el error en la respuesta del chatbot presentado por: Rollo Carpenter, llamado: "Joan" quien resultó vencedor en el certamen del año 2006.

human: Do you read books?

chatbot: Sure I read books.

human: Who is your favourite author?

chatbot: I don't know. There are so many. Who's your favourite author?

human: I like Philip Roth

chatbot: Are they scary?

Aquí el chatbot da una respuesta no satisfactoria a la frase del usuario "I like Philip Roth", no considera la frase : "my author favorite is Philip Roth".

Ejemplo 3)

A continuación se muestran tres problemas juntos: de resolución de referentes, de referencias inexistentes y elipsis; en el dialogo entre del chatbot presentado por: Rollo Carpenter, llamado: "Joan" quien resultó vencedor en el certamen del año 2006, y el Juez 3 del certamen.

chatbot: I am familiar with her writings.

human: Really like what?

chatbot: It used to be in a smallish building. It is now in a part of a larger building. Someday they hope to build a new library

human: Is it safe?

chatbot: Probably.

human: Is it safe?

chatbot: Do you think it's safe?

human: Is it safe?

chatbot: Of course.

human: Is it safe?

chatbot: Yes.

human: Are you sure?

chatbot: It's easy to be sure about something like that when you're 26 years old!

El dialogo que se presenta arriba presenta demasiadas dificultades juntas para un chatbot, aunque resulte trivial para un ser humano. Primero se tiene la interrogación del usuario: "Is it safe?" la cual presenta una referencia anafórica, aunque no está muy claro a que hace referencia el pronombre "it" podría ser "larger building", en dicho caso la frase quedaría como: "Is [the] larger building safe ?". Ante dicha interrogación, el chatbot responde de forma genérica: "Probably". Luego el usuario vuelve a preguntar nuevamente por tres veces: "Is it safe?", con lo cual queda claro de que la referencia del pronombre "it" no existe, cualquier persona respondería ante la misma situación con un interrogante del estilo: "What do you mean with that question?"

Por último el usuario pregunta: "Are you sure?", aquí hay una elipsis ya que está implícito en la oración "it is safe", si se reformulase la sentencia para eliminar la elipsis la frase quedaría como: "Are you sure that it is safe?". Sin embargo el chatbot devuelve una respuesta genérica que solo hacer referencia a la palabra "sure"

Ejemplo 4)

A continuación se muestra un problema de resolución de referentes y el error en la respuesta del chatbot presentado por: Richard Wallace, llamado: "ALICE" quien resultó vencedor en el certamen del año 2004. (Extracto de la conversación el Juez 2)

human: visiting Dr Loebner, but I live in New Jersey

chatbot: I have no idea where that is. Tell me a little about your life in visiting Dr Loebner but you live in New Jersey.

human: it's right across the river from here

chatbot: Do you experience any effects of global warming there?

human: oh sure, all the time

En la frase del usuario: "it's rigth across the river from here.", el pronombre "it" está haciendo referencia a "Dr. Loebner('s house)", si la frase se reformulara para eliminar la anáfora quedaría como: "Dr Loebner is rigth across the river from here". Al tener acceso al código y a la base de datos de ALICE es fácil ver que el patrón de coincidencia utilizado fue: "IT IS * HERE", (siendo * un wildcard), la respuesta dada por el chatbot fue una respuesta genérica.

En este caso en particular se presenta otro problema de la lengua y del procesamiento de lenguaje natural en general, no necesariamente relacionado con la interpretación pragmática; la metonimia, La metonimia es una figura del habla en la cual se emplea un objeto para sustituir otro objeto. Aquí "Dr Loebener" en verdad es una figura que se usa en "reemplazo" de "la casa del Dr Loebener".

Ejemplo 5)

A continuación se muestra un problema de resolución de referentes y el error en la respuesta del chatbot presentado por: Richard Wallace, llamado: "ALICE" quien resultó vencedor en el certamen del año 2004. (Extracto de la conversación el Juez 1)

human: What is your favorite operating system?

chatbot: Linux is the best operating system.

human: What do you like about it?

chatbot: it? What do I like about it?

human: What do you like about Linux?

chatbot: The special effects are the best.

En este ejemplo, queda claro para cualquier persona que en la frase: "What do you like about it?", "it" es una referencia a "Linux". El chatbot no puede resolver esta anáfora y responde a la pregunta textual: "What do you like about it?", el usuario elimina la anáfora y vuelve a hacer la pregunta: "What do you like about Linux?" en este caso el chatbot da una respuesta un poco más satisfactoria, aunque no del todo.

El patrón de coincidencia que encuentra AIML para la última frase de entrada del usuario es: "WHAT DO YOU LIKE ABOUT *", con lo que la respuesta es bastante genérica. Sería igual para: "What do you like about cookies?" por ejemplo.

IV. Solución propuesta

En el presente capítulo se describe la solución propuesta a los problemas planteados en el punto anterior; En el punto 4.1 se esboza el aspecto general de la solución y la forma en que cada una de las tecnologías propuestas se ajustaría a un requerimiento dado. En el punto 4.2 se analizan las diferencias entre las líneas de dialogo: usuario-chatbot consideradas correctas o coherentes (casos favorables) y aquellas consideradas incorrectas (casos desfavorables) que será de mucha importancia para entender el punto 4.3, en donde se describe como se utilizará una RNA para clasificar los casos. En el punto 4.4 se explica la necesidad de utilizar una base de datos relacional en lugar de archivos XML para gestionar la base de datos de conocimiento de ALICE. Por último los puntos 4.5, 4.6 y 4.7 explican los tres mecanismos propuestos para mejorar la forma de responder de un chatbot basado en AIML teniendo en cuenta la interpretación pragmática.

A. Esbozo general de la solución

La premisa general de la investigación es mejorar un chatbot, o una tecnología de diseño de chatbots, que esté actualmente en el *estado del arte (o de la cuestión)* y no crear un nuevo chatbot desde cero. Para ello se utilizará como punto de partida el *lenguaje* AIML y la base de datos de conocimiento de ALICE, ambos libres y disponibles al público general.

El primer paso para encarar la solución consiste en separar los casos *favorables* de los desfavorables (ver punto siguiente) de forma que para todas aquellas frases de entrada para las cuales ALICE ha dado buenos resultados no haya modificaciones y para todas aquellas frases para las cuales ALICE da respuestas no satisfactorias, aún cuando estas coinciden con algún patrón de la base de datos de conocimientos, se activen mecanismos que revisen la información de contexto, dada solamente por todas las líneas de dialogo anteriores, y la utilicen para convertir el caso desfavorable en un caso favorable.

Para realizar la clasificación se utilizará una red neuronal artificial de tipo *Back Propagation*, luego para convertir los casos desfavorables en casos favorables se utilizarán tres mecanismos diferentes, el primero y más importante es un algoritmo de resolución de referentes anafóricos (*RAP*, descripto en 2.42), el segundo es un procedimiento simple para reformular preguntas cerradas como sentencias declarativas, ejemplo: "Do you like movies?" será reformulada como "I like movies" si el usuario hubiese ingresado "yes" como respuesta. El tercer mecanismo es un procedimiento que se activará cuando los demás hayan fallado, se trata de un procedimiento que busca referentes insatisfechos y pregunta por su significado. Por ejemplo si el algoritmo de resolución de referentes anafóricos, falló al tratar de identificar el pronombre "her", en la oración: "Do you like her?" este procedimiento hará que el chatbot devuelva una pregunta como la siguiente: "Who is her?"

B. Distinción entre casos favorables y casos desfavorables

Tomando al chatbot ALICE, como modelo inicial a mejorar, lo primero que se propone hacer es clasificar los casos en favorables y desfavorables. Es decir, para todas aquellas frases de entrada de un usuario, en el conjunto universo de frases posibles, determinar cuáles obtendrán por parte del chatbot, una respuesta satisfactoria (una respuesta que deje al interlocutor satisfecho en igual forma que si hubiese respondido un ser humano en iguales condiciones) y cuáles no.

Planteado en términos absolutos el problema es inabarcable por varia cuestiones a saber:

1) Casos limitados en la base de datos de conocimiento de ALICE
2) Restricción del idioma (en esta investigación se restringe al idioma inglés)
3) Frases inválidas, mal formadas o con errores sintácticos, ortográficos u otros.
4) La imposibilidad de saber de forma objetiva que espera un ser humano al hacer una pregunta. ¿Cómo determinar de forma objetiva que una respuesta es correcta desde el punto de vista de la coherencia de la oración y no de su significado?

Con la intensión de acotar el alcance al problema anterior de clasificación se analizaron los casos favorables en los diálogos de ALICE. Se muestran a continuación algunos ejemplos que provienen del certamen Loebener de 2004 [21].

Caso 1)

Judge 1: Do you like movies?

Alice: Yes I love film, especially science-fiction and comedy.

Caso 2)

Judge 1: What is your favorite movie?

Alice: My favorite movie is Teknolust. Have you seen it?

Caso 3)

Judge 1: Do you have a pet?

Alice: No but I like cats very much.

Caso 4)

Judge 1: Please, call me Ryan.

Alice: Thank you for your request. I am very pleased to meet you Ryan.

Lo que tienen en común estos casos, (como muchos otros casos favorables analizados) es que la frase de entrada, lo que en estos diálogos está representado por lo que dijo "Judge 1", no presentan referencias, ni elipsis, ni ambigüedades léxicas (evidentes) tampoco son frases monosilábicas o respuestas cortas a alguna pregunta anterior.

De forma intuitiva se puede pensar en que este tipo de frases podrían ser las iniciadoras de un dialogo con una persona ya que al iniciar un dialogo hay muy poca información de contexto. Si el dialogo es de forma escrita y los interlocutores están en habitaciones separadas, la información contextual es aún menor. Es decir, una persona "A" podría decirle a otra "B" como primera frase de una conversación:

"Do you like movies?", "What is your favorite movie?" o "Please, call me Ryan" y la persona "B" sería capaz de responder de forma satisfactoria, casi tan satisfactoria como si la misma frase estuviese ubicada a la mitad de un dialogo. No quiere decir que sean frases para iniciar una conversación, por supuesto, ya que un saludo de despedida como "See you soon" también estaría dentro de este conjunto de casos.

Sin embargo lo anterior no es completamente cierto ya que en un contexto demasiado forzado podrían dejar de ser casos favorables para un chatbot AIML, por ejemplo si el usuario dijese antes de "Do you like movies?", "there are new candies called movies" el chatbot daría la misma exacta respuesta: "Yes I love film, especially science-fiction and comedy." Tampoco serían validos en los casos en donde un usuario emplease recursos como la ironía o eufemismos, pero esta clase de problemas escapan en parte a lo que es la interpretación pragmática y al objetivo de este trabajo de estudio.

A pesar de que un contexto forzado podría invalidar los casos, estos serían validos en la mayoría de los diálogos con lo cual la definición de "caso favorable" pasa a ser una definición probabilística que podría definirse como sigue:

El caso "A" pertenece al conjunto de casos favorables si en el X % de los diálogos, cuando un usuario ingresa la frase: "A" obtiene por parte el chatbot una respuesta coherente que satisface sus expectativas tal y como lo haría la respuesta de una persona en circunstancias similares.

Donde X es un número menor que 100, aleatoriamente alto. La definición anterior es una definición practica que es cómoda a los efectos de clasificar frases de entrada como casos favorables para el objeto de estudio de esta investigación, sin embargo la tarea de clasificar casos utilizando la definición anterior es casi imposible debido a que habría que evaluar de forma subjetiva cuando una frase obtuvo una respuesta satisfactoria y cuando no en todos los diálogos posibles (o bien en algún número suficientemente grande de diálogos.)

Volviendo al ejemplo de cómo se podría clasificar de forma intuitiva un caso; así como la persona "A" al iniciar el dialogo con la persona "B" podría utilizar frases como las anteriores, no podría hacerlo con frases como: "Do you like her?", "What's your favorite?", "yes" o incluso frases como "I'm opened the door" que contienen poca información en el sentido que le da Shannon en [Shannon 1948] y esperar una respuesta igual a la que recibiría si dichas sentencias se hallasen a mitad de un dialogo en donde ya se "sabe" a quien hace referencia el pronombre "her" o que se está hablado de "movies" o que "yes" responde a determinada pregunta o que "door" fue abierta.

De alguna forma, los casos favorables son aquellos en donde la frase de entrada no depende del contexto (en la mayoría de los casos) y los casos desfavorables son aquellos en donde la frase de entrada depende del contexto. Los primeros están compuestos por frases que serán llamadas a lo largo de esta investigación: "independientes del contexto" y los segundos de frases que serán llamadas "dependientes del contexto", aún cuando no exista en la base de datos de conocimiento del chatbot un patrón de coincidencia para la frase. (Se estaría en este caso ante otro problema relacionado con la completitud de la base de datos de conocimiento.)

Los casos serán clasificados según la categoría anterior utilizando una red neuronal artificial.

C. Clasificación de los casos usando una RNA

Parte importante de esta investigación es crear una red neuronal artificial que sea capaz de detectar cuando una frase es dependiente o independiente del contexto. La otra parte es la transformación de una frase dependiente del contexto en una frase independiente del contexto, que es aquella, como fue explicado en el punto 4.1, que constituye un caso favorable para un chatbot basado en AIML.

Para la construcción de la RNA se propone crear, primeramente, un conjunto de entrenamiento en donde haya frases en inglés y por cada frase se indique si es o no dependiente del contexto. Las frases en inglés serán transformadas en un mapa que será utilizado como entrada de la RNA.

El mapa se generará no a partir de las frases en sí, sino a partir de los tipos básicos de palabras que la conforman, el orden de estas en la oración, si la oración es una interrogación o no y el tiempo verbal en el cual está formulada la oración. Se hará especial hincapié en los distintos tipos de pronombres, el resto de las palabras serán clasificadas en alguno de los siguientes tipos básicos: sustantivo (*noun*), verbo (*verb*), adjetivo (*adjective*), adverbio (*adverb*), artículos y palabras auxiliares (*aux*)

El tipo de arquitectura de red neuronal escogido es el de *Back Propagation*, descripto en el punto 2.4.1 debido a que entre los tipos de redes con aprendizaje supervisado esta es una

de las más exitosas, utilizada en casos de reconocimiento de texto, de voz, clasificación de patrones, etc.

Durante la fase de entrenamiento se creará una red con una capa oculta ya que de esta forma se pueden clasificar elementos encerrados en regiones polinomiales convexas del espacio y otra con dos capas ocultas que permitirá clasificar elementos encerrados en conjuntos de formas arbitrarias sobre el espacio dado ya que no conocemos la distribución espacial de las frases dependientes e independientes del contexto (según los mapas utilizados). La capa de salida tendrá una única neurona ya que se espera un solo valor *boleano*: verdadero o falso, 1 ó 0 que indicaría si es independiente del contexto o no. Las capas intermedias serán creadas con una cantidad de neuronas igual a la mitad de las utilizadas en la primera capa, al menos en su configuración inicial. La capa de entrada tendrá tantas neuronas como los mapas propuestos necesiten.

Se proponen dos mapas posibles como *input* de la red neuronal, ambos se describen en los puntos siguientes.

1) Mapa 1 propuesto como input de la RNA

Sobre la frase original en inglés se realizan 3 procesos diferentes:

I) Se reemplaza cada palabra por su tipo: sustantivo (*noun*), verbo (*verb*), adjetivo (*adjective*), adverbio (*adverb*), artículos y palabras auxiliares (*aux*). (Por palabra se entiende una o más cadenas de caracteres separadas por espacio que juntas tienen un significado univoco distinto del que tienen por separado, ejemplo: "Buenos Aires" es una palabra compuesta por dos cadenas de caracteres). Cada tipo de palabra tiene un valor numérico asociado.

II) Se aplica un procedimiento para detectar si la frase es una interrogación o no. (Aunque no tenga el signo de interrogación)

III) Se aplica un procedimiento que mediante una serie de expresiones regulares detecta el tiempo verbal de la oración.

Una vez hecho lo anterior se crea un vector de números en punto flotante de 15 posiciones que será finalmente el *input* de la RNA. Se instancian todas las posiciones en cero.
Luego en la posición 0 del vector se marca si la frase es una interrogación con un uno (1) o con un cero (0) sino. La posición 1 del vector indica el tiempo verbal, el cual puede tomar uno de los siguientes valores: 0.0; 0.5; 0.7; -1.0; -1.3; 0.8; 0.6; -1.4; -1.5; 1.1; 1.2; 1.3; 1.5; 1.9 para los tiempos verbales respectivos (en inglés): *infinitive, simple present, present continuous, past, past continuous, present perfect, present perfect continuous, past perfect, past perfect continuous, simple future, future continuous, future perfect, future perfect continuous, conditional*. El resto de las posiciones del vector representan en orden, a los diferentes tipos de palabras que conforman la oración (las primeras 13 palabras), según su valor numérico. Los tipos de palabras y sus valores numéricos asociados son:

auxiliary words 0.1
noun 1.9
adjetive 1.3
indicative 2.3
intterrogative pronouns 2.0
verbs 1.5
personal pronouns fisrt person: 2.5
personal pronouns third person: 3.0
personal pronouns second person: 3.5

El mapa puede ser representado de forma esquemática como se muestra en la Figura 7

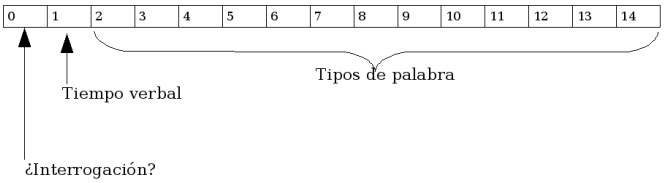

Fig. 7. Mapa 1

2) Mapa 2 propuesto como input de la RNA

Para la generación del *input* 2 de la RNA utilizada se cambió el mapa de entrada para hacerlo más sensible. Lo que se hizo fue aumentar el número de neuronas de entrada. Este nuevo mapa reserva una neurona especial para un valor de entrada que indica si la frase es o no una pregunta, al igual que en el caso anterior se realizan las siguientes acciones sobre la frase de entrada:

I) Se reemplaza cada palabra por su tipo: sustantivo (noun), verbo (verb), adjetivo (adjective), adverbio (adverb), artículos y palabras auxiliares (aux).

II) Se aplica un procedimiento que detecta si la frase es una interrogación o no.

Pero no se realiza la detección de tiempo verbal, se eliminó la neurona que respondía ante el tiempo verbal encontrado identificado en la oración.

El resto del mapa de entrada se definió como una matriz de 15 x 5 números de punto flotante, en donde la longitud, 15, representa una a una las primeras 15 palabras de la frase de entrada y el ancho o profundidad: 5, indica según la posición que tome un valor distinto de cero, que tipo de palabra es. En donde la posición 0 del vector vertical toma un valor distinto de cero si la palabra es auxiliar o bien si esta es un pronombre interrogativo (*interrogative pronouns*). En el caso de una palabra auxiliar, el valor que toma es de 0.2, en el caso de un pronombre interrogativo es de 3.0 (en este caso se verá reforzado además por la neurona que indica que es una pregunta.) La posición 1 del vector vertical toma valor igual a 1.0 si la palabra es un verbo (verb*)* o bien si es un adverbio (*adverb*). La posición 2 toma un valor igual a 1 si es un adjetivo (*adjective*). La posición 3 toma un valor igual a 1 si es un sustantivo (*noun*) común o igual a 2 si es un sustantivo propio. Por último la posición 4 toma un valor distinto de cero si la palabra es un pronombre personal (personal *pronoun*) o un pronombre demostrativo (*demostrative pronoun*). En estos casos puede tomar diferentes valores, tomará el valor 1 para pronombres personales en primera persona, 2 para pronombres personales en tercera persona y 3 para pronombres personales en segunda persona. Para pronombres demostrativos toma el valor 2.5.

Esquemáticamente el mapa quedó como se muestra en la Figura 8:

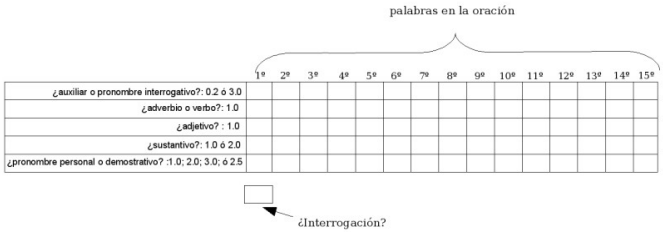

Fig. 8. Mapa 2

3) Pasos para la clasificación

La clasificación de casos incluye analizar todos los patrones (*patterns*) de cada una de las categorías (*category*) de ALICE para indicar si la frase contenida en el *tag pattern* es o no independiente del contexto.

Como primera medida se clasificarán todas las categorías (*category*) cuyos *patterns* esten conformados por frases que utilicen *wildcards* como dependientes del contexto ya que a priori no se puede saber qué tipo de frase de entrada de usuario coincidirá con un patrón con *wildcards.*

Como segunda medida se clasificarán como independientes del contexto a todas aquellas categorías (*category*) que utilicen el *tag* "that", ya que si bien el *pattern* en sí es dependiente del contexto, la *category* ya lo tiene contemplado.

Como tercera medida se utilizará la RNA previamente entrenada, para clasificar todas las categorías restantes.

El nuevo campo a agregar en cada *category*, para indicar si es o no un caso favorable se llamará "can_answer" ya que este nombre es más general. (Una *category* con *tag that* es dependiente del contexto, pero puede ser utilizada para responder ya que conforma un caso favorable.)

Una vez que el sistema esté funcional, trabajará inicialmente de forma idéntica a como lo hace ahora AIML, es decir seguirá de forma esquemática los siguientes pasos:

1) Esperar una frase de entrada del usuario
2) Buscar una *category* cuyo *tag pattern* coincida lo mejor posible con la frase anterior.
3) Tomar la frase contenida en el *tag template*.

Pero al llegar a este punto, revisará el nuevo campo "can_answer". Si este campo es uno (1) el sistema procederá de la forma usual, si es cero utilizará la RNA para clasificar la frase de entrada del usuario. Si el resultado es (1): independiente del contexto, procederá de la forma usual. Si es cero (0): dependiente del contexto aplicará los mecanismos antes descriptos para convertir la frase del usuario en una frase independiente del contexto.

4) Utilización de redes neuronales artificiales autoorganizadas (SOM)

La red autoorganizada: *SOM* del modelo propuesto por Kohonen será tenida en cuenta como un clasificador de frases que podría ser utilizada previo al uso del perceptrón. Se utilizará la herramienta indicada en [17]. Una forma común de reducir el error cuadrático medio del algoritmo *Back Propagation*, es dividir el conjunto de pruebas original en subconjuntos donde los elementos agrupados presenten similares características. Una forma de lograr estos subconjuntos es mediante la clasificación realizada a través de redes autoorganizadas.

Los *inputs* que tomaría, eventualmente, esta red serían los mismos propuestos para la red perceptrón.

D. Implementación de AIML sobre mySQL

Para simplificar la tarea de análisis y clasificación de los *tags AIML*, la totalidad de la base de datos de conocimiento de ALICE será migrada a una base de datos relacional mySQL.

Durante todo el proceso de pruebas y correcciones se utilizará una implementación de AIML sobre mySQL, la cual reemplazará al algoritmo de *pattern-matching* de los intérpretes AIML por el propio *engine* mySQL. Dicha implementación será respaldada por una serie de pruebas que garanticen que la implementación sobre mySQL funcione de igual forma que el intérprete "program-D" utilizado en la actualidad en el sitio de ALICE *online.*

E. Resolución de referentes

La resolución de referentes es el mecanismo más importante con el que se cuenta para convertir los casos desfavorables en casos favorables, dicho con la nomenclatura de esta investigación convertir las frases dependientes del contexto en frases independientes del contexto. Para ello se utilizará el algoritmo: RAP (Resolution of Anaphora Procedure.) El cual ha dado buenos resultados según [7]: 74 % de tasa de éxito en la resolución de anáforas ínter-sentencias y 89 % de tasa de éxito en la resolución de anáforas intra-sentencias.

Este algoritmo entraría en ejecución luego de que una frase de entrada del usuario haya sido detectada como caso desfavorable y la RNA haya devuelto el valor cero al procesar el mapa asociado a dicha frase. Entonces el algoritmo *RAP* tomaría como *input* las últimas 10 líneas del dialogo y la última frase del usuario. Buscaría en la frase alguna referencia anafórica y trataría de vincularla con algún sustantivo presente en las líneas previas del dialogo. Como salida, el algoritmo devolvería la frase del usuario reformulada como una nueva oración independiente del contexto cuyo significado es el mismo.

De forma ilustrativa se propone el siguiente ejemplo: supóngase la siguiente conversación con el chatbot ALICE, utilizando su base de conocimientos y su funcionamiento actual:

Human: Tell me about Richard Wallace

ALICE: Are you asking about my botmaster?

Human: He is a great inventor

ALICE: Maybe you should tell him how you feel about him.

La frase de entrada: "He is a great inventor", coincide con el patrón "HE IS A GOOD %", con lo cual la respuesta dada por el chatbot no es más que una respuesta genérica que no considera a Richard Wallace como sujeto de la oración.

Con los cambios propuestos implementados la frase: "He is a great inventor" sería clasificada como dependiente del contexto por la RNA, y lo es ya que el pronombre "he" es una referencia anafórica por resolver. Se aplicaría luego el algoritmo *RAP*, que utilizaría como *input* las dos líneas anteriores del dialogo y la frase actual para devolver lo siguiente:

"Richard Wallace is a great inventor".

Esta frase sería utilizada como una nueva entrada de usuario por el chatbot la cual coincidiría con otro patrón:

"RICHARD WALLACE IS %". Este patrón considera a "Richard Wallace" como sujeto de la oración con lo cual devolvería una respuesta más coherente en relación al dialogo.

F. Reformulación de respuestas a preguntas cerradas

Este es otro mecanismo capaz de convertir respuestas dependientes del contexto en respuestas independientes del contexto pero su algoritmia es mucho más sencilla que la del *parser* RAP aunque su alcance es también mucho más limitado.

Fue concebido como una forma sencilla de resolver un problema harto común en los diálogos de los chatbots. En particular se notó que ALICE cuenta en su base de datos de conocimiento con respuestas (más correctamente hablando frases contenidas por el *tag template*) las cuales no son más que preguntas cerradas, como por ejemplo:

"Do you like chocolate ?"
"Do you like computers?"
"Are you a woman ?"
"Are you an english teacher?"

Este tipo de interrogantes suelen ser respondidos mediante un sí (*yes*) o un no (*no*), o bien mediante algún sinónimo más coloquial: "sure", "of course", "no way", etc. En la base de datos de ALICE no hay entradas para cada una de estas posibles respuestas que utilicen el *tag that* para remarcar la pregunta anterior; y agregarlas, por otro lado, sería poco eficiente ya que implicaría repetir muchas entradas. Sin embargo sí cuenta con patrones como "I am a woman" o "I like chocolate". En base a estas observaciones se pensó en una forma de convertir las frases como: "yes", "no", "sure", etc. en frases independientes del contexto como "I like chocolate" siempre y cuando dichas frases sean respuestas a preguntas cerradas.

De forma ilustrativa se propone el siguiente ejemplo: Supóngase la siguiente conversación con el chatbot ALICE, utilizando su base de conocimientos y su funcionamiento actual:

Human: I hate cats
ALICE: Do you like dogs?
Human: yes
ALICE: Hmm.

En este ejemplo la frase de entrada "yes" coincidió con el patrón "yes", para el cual hay asignada una respuesta genérica, en este caso "Hmm". Si el cambio propuesto estuviese implementado, la frase de entrada "yes" habría sido detectada como dependiente del contexto por la RNA , luego se hubiese procedido a verificar si la frase anterior del chatbot era una respuesta cerrada; entonces las frases "yes" y "Do you like dogs?" hubiesen sido combinadas y reformuladas como: "I like dogs", una frase independiente del contexto. Dicha sentencia sería entonces utilizada como frase de entrada por el chatbot (en lugar de "yes") y habría coincidido con un patrón idéntico "I LIKE DOGS" cuya respuesta asociada (contendido el *tag template*) es: "Which breed is your favorite?"

G. Preguntas por referencias no encontradas

Se pretende construir un subsistema que sirva como *backup* para los casos en los cuales el *RAP* no logre resolver la referencia anafórica. El funcionamiento es simple de explicar, cuando una frase dependiente del contexto que presente un pronombre no logre ser reformulada, este subsistema detectará dicho pronombre y formulará una frase de interrogación que será devuelta al usuario. En dicha respuesta se pedirá la información faltante.

De forma ilustrativa se propone el siguiente ejemplo: Supóngase la siguiente conversación con el chatbot ALICE, utilizando su base de conocimientos y su funcionamiento actual:

Human: He is a great inventor
ALICE: Maybe you should tell him how you feel about him.

Esta conversación es idéntica a la presentada en el punto 4.4, solo que en este dialogo no se hace referencia a Richard Wallace sino que empieza directamente con la frase: "He is a great inventor". Se puede ver que la respuesta de ALICE tendría sentido si se supiese de quien se está hablando, es decir, a quien hace referencia la palabra "he". Claramente un interlocutor humano no respondería de esa forma sino que preguntaría por la referencia perdida, una respuesta "más humana" sería: "Who are you taking about?"

Suponiendo que estuviesen implementados los cambios propuestos el algoritmo *RAP* se aplicaría sobre la frase: "he is a great inventor" pero no podría encontrar la referencia anafórica de "he" sencillamente porque no existe. Entonces actuaria la "última línea de defensa", que es este subsistema, el cual buscaría la palabra "clave" de una lista de palabras posibles y la utilizaría para formular una pregunta, que podría ser: "Who is he?".

Cabe destacar que este subsistema, o procedimiento según como termine siendo implementado no significa ninguna novedad, AIML mismo o incluso los chatbots basados en ELIZA son capaces de buscar una palabra "clave" en una frase y a partir de ella formular una pregunta. Lo que se presenta aquí como novedoso es la forma y la situación en la cual se lo pretende emplear.

V. DESARROLLO DE LA SOLUCIÓN

El objetivo de este apartado es el análisis del conjunto concreto de necesidades para proponer una solución a corto plazo, que tenga en cuenta restricciones técnicas y operativas. La solución obtenida como resultado de este estudio debe ser la definición de un proyecto que afecte a sistemas existentes o bien cree uno nuevo. Se identificarán, para ello los requisitos que se han de satisfacer y se estudiará la situación actual.

A. Establecimiento del alcance del sistema

En este apartado se estudiará el alcance de la necesidad planteada realizando una descripción general de la misma. Se determinarán los objetivos y se iniciará el estudio de los requisitos. Se analizarán además las posibles restricciones, tanto generales como específicas, que puedan condicionar el estudio y la planificación del trabajo de investigación.

1) Resumen de la propuesta de investigación

La solicitud en este caso está determinada por la propuesta de investigación, esta propuesta sugiere la implementación de una mejora en los sistemas actuales conocidos como *chatbots*.

Los chatbots actualmente no logran reproducir el comportamiento humano en una conversación ni dar respuestas satisfactorias que puedan ser consideradas coherentes. Solo responden de manera satisfactoria a determinadas frases. Ejemplos de estas conversaciones pueden obtenerse en [9], [10] y [11].

El estudio de las transcripciones de las conversaciones sostenidas entre estos sistemas y un individuo dado muestran

diversos problemas, entre los cuales se encuentra la falta de evaluación del contexto (de la conversación) al responder.

La mejora propuesta consiste en modificar un chatbot existente de tal manera que pueda responder a una frase de un individuo teniendo en cuenta información de contexto, entendiendo por contexto frases mencionadas anteriormente en una misma conversación.

2) Descripción del sistema a construir y objetivos

El sistema a construir será un chatbot que se encuentre dentro del *estado del arte* (o de la cuestión) de este tipo de programas, es decir que pueda responder a frases de entrada al menos tan satisfactoriamente como cualquier otro chatbot reconocido. Pero este chatbot además deberá de poder responder correctamente en situaciones en las cuales un chatbot actual no lo hace.

Se tomará como punto de partida para el desarrollo de dicho sistema, un chatbot existente cuyo código fuente y base datos se encuentren disponibles y en lo posible se disponga de documentación acerca de su funcionamiento. Se estudiarán los casos en los cuales el chatbot responde de manera no satisfactoria por no tener en cuenta el contexto. Se evaluarán posibles mejoras para dichos casos y se aplicarán sin modificar el comportamiento para los casos en los cuales el sistema ya funciona adecuadamente.

3) Diagramas de flujo

En la Figura 9 que se muestra a continuación, está representado, mediante un diagrama de flujo el funcionamiento de un chatbot actual. En el siguiente diagrama se representa de manera esquemática como afectarían los cambios propuestos a dicho sistema.

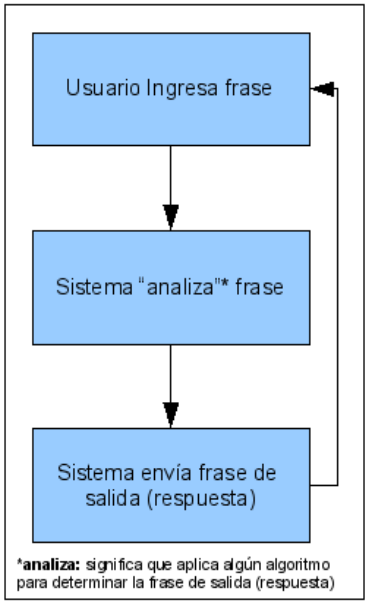

Fig. 9. Flujo, chatbot actual

A partir de la observación del segundo diagrama se desprende que las modificaciones implican la creación de un sub-sistema capaz de decidir cuando una frase de entrada de un usuario dado puede ser "respondida" correctamente por el chatbot y cuando es necesario buscar información de contexto (en frases anteriores) para responder correctamente. En otras palabras dicho sub-sistema debería de poder determinar si una frase es dependiente o independiente de su contexto. Este sub-sistema estaría actuando también como un filtro que permite dividir el funcionamiento del chatbot en dos, la parte o los casos satisfactorios y los no satisfactorios. A continuación entra en juego un segundo sub-sistema que tomando una frase de entrada que está dentro del conjunto de los casos no satisfactorios le agrega información de contexto que obtiene de frases anteriores, cambiando de esta manera la frase del usuario. Siguiendo el flujo, si el sub-sistema encontró la información faltante y pudo *reformular*, con ella la frase, esta vuelve al flujo principal del chatbot.

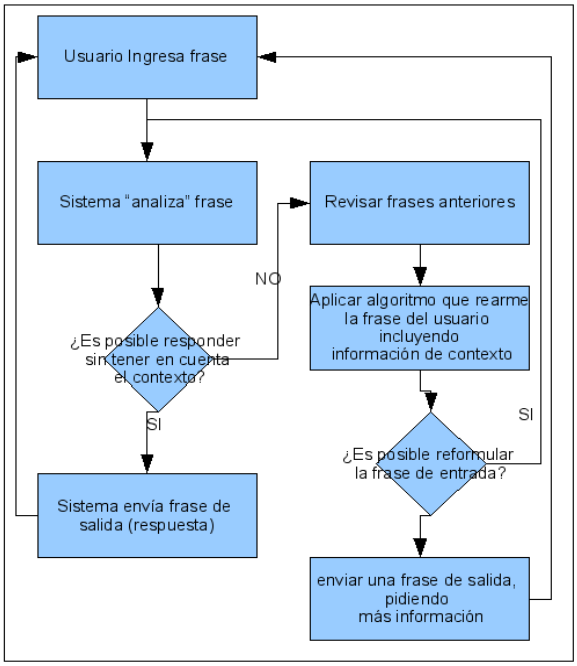

Fig. 10. Flujo, chatbot modificado

En este caso el chatbot podrá responder de manera satisfactoria (o no, pero en caso de no hacerlo el problema no debería de ser la evaluación del contexto sino otro, como falta de información en su base de datos.) Para el caso en el cual el sub-sistema anterior no fue capaz de encontrar la información de contexto faltante, o de *reformular* la pregunta se enviará una frase de salida al usuario, pidiendo la información de contexto faltante.

4) Requisitos

A partir del análisis anterior es posible identificar una serie de requisitos mínimos, al menos en una primera estimación, necesarios para llevar adelante el desarrollo del sistema propuesto:

- El código fuente y base de datos de un chatbot que esté en el *estado del arte* y que se pueda modificar, el mismo servirá como punto de partida.
- Desarrollar un sub-sistema capaz de decidir si una frase de entrada puede ser respondida de manera correcta por el chatbot anterior o si es necesario información de contexto para ello, es decir que verifique si una frase dada es independiente del contexto o no. (sub-sistema detector)
- Desarrollar un sub-sistema que pueda convertir una frase dependiente del contexto por otra que mantenga su sentido y significado pero sea independiente del contexto. Utilizando para ello frases anteriores de un mismo dialogo. (sub-sistema conversor)

5) Especificación del alcance del sistema

A partir de los objetivos y requisitos planteados en los ítems anteriores, es posible elaborar una primera serie de tareas necesarias para cumplir dichos objetivos:

I. Investigación de los algoritmos y/o modelos actuales que permiten dilucidar el contexto en una conversación y estudiar como son aplicados en modelos de chatbots.

II. Selección de un chatbot, comprensión y entendimiento de su funcionamiento.

III. Definición y especificación de mejoras a realizar en el modelo de chatbot escogido para lograr que este adapte su comportamiento a partir de la evaluación del contexto.

IV. Construcción e implementación de un sub-sistema capaz de determinar si una frase es independiente del contexto o no. (sub-sistema detector)

V. Construcción e implementación de un sub-sistema capaz de convertir una frase dependiente del contexto en una frase independiente del contexto. (sub-sistema conversor)

VI. Implementación del modelo de chatbot con las mejoras propuestas con la integración de los sub-sistemas necesarios.

VII. Pruebas de integración, desarrollo de una interfaz de usuario para que cualquier persona pueda interactuar con el nuevo chatbot.

VIII. Corroboración de los supuestos teóricos, corridas de prueba, comparación con los modelos tradicionales y correcciones necesarias.

TABLA VIII. Estimación PERT

Tarea \ Estimación (días)	O	M	P	PERT
1: investigación	6	12	20	12,33
2: Selección chatbot	5	7	10	7,17
3: Definición mejoras	10	15	20	15
4: sub-sistema detector	15	17	21	17,33
5: sub-sistema conversor	12	15	25	16,17
6: integración sub-sistemas y chatbot	5	6	7	6
7: Pruebas integración/interfaz	5	6	7	6
8: Pruebas generales/correcciones	4	7	8	6,67

.

La tabla anterior proporciona una noción aproximada del esfuerzo requerido para la elaboración del sistema final en días/hombres. Pero para saber el tiempo de entrega de cada tarea, y por ende el tiempo que se demorará en la finalización del sistema es necesario conocer la cantidad de horas por semana que se dedicarán al desarrollo y confección del sistema.

Días por semana: 2 (un total de 16 hs. por semana en promedio.) En la siguiente tabla se muestra el tiempo de entrega estimado.

El número de días fue redondeado ya que esta primera estimación es grosera y el tiempo de dedicación semanal puede variar.

TABLA IX. *Tiempo de entrega de cada tarea*

Tarea	entrega en días
1: investigación	40
2: Selección chatbot	24
3: Definición mejoras	50
4: sub-sistema detector	60
5: sub-sistema conversor	55
6: integración sub-sistemas y chatbot	20
7: Pruebas integración/interfaz	20
8: Pruebas generales/correcciones	20
Total:	289

Colocando en un diagrama de Gantt los números anteriores, para un solo recurso asignado a todas las tareas, obtenemos un gráfico como el siguiente:

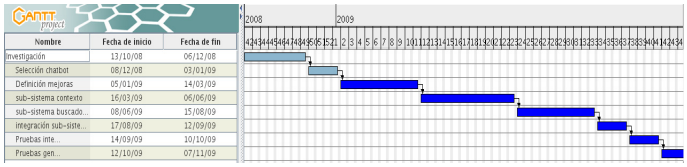

Fig. 11. Gantt de proyecto

B. Estudio de la situación actual

En la actualidad existe una cantidad demasiado grande de chatbots, los cuales no están catalogados, descriptos ni enumerados en ningún documento confiable y no siempre es posible describir el funcionamiento de estos porque simplemente no hay documentación disponible ni acceso al su código fuente. De todos los chatbots, solo interesan al objeto de este proyecto, aquellos que estén entre los mejores según algún criterio medianamente objetivo y que además su código fuente y base de datos estén disponibles.

La selección de dichos sistemas será hecha en base a tres criterios:

1. Aquellos chatbots que hayan sido finalistas en el concurso llamado "The Loebner Prize in Artificial Intelligence", el cual es una primera aproximación al Test de Turing [18]. El concurso es anual y se presentan, en EE.UU., diferentes chatbots que funcionan en inglés; gana el que se aproxime más a un ser humano en su modo de responder.
2. Chatbots o *engines* de chatbots que sean utilizados de manera comercial o productiva de manera exitosa.
3. Aquellos chatbots cuyo código fuente y base de datos estén disponibles para ser examinadas y modificadas.

En base a los requisitos 1 y 2 se confeccionó una lista de chatbots:

Ultra Hal: Este chatbot ganó el primer lugar en el concurso Loebner del año 2007. [9]

disponible: http://www.zabaware.com/webhal/index.html

ALICE: ALICE está desarrollado sobre AIML y fue ganador del premio Loebner en tres oportunidades.

disponible: http://www.pandorabots.com/pandora/talk?botid=f5d922d97e345aa1

Jabberwacky: Jabberwacky fue ganador del premio Loebner en dos oportunidades.

disponible: http://www.jabberwacky.com/

ELIZA: Eliza fue el primer chatbot exitoso construido, extendiendo a Eliza se construyó "PC therapist" un chatbot que fue ganador en los primeros tres años del concurso Loebner. Y luego volvió a ganar en el año 1995.

Disponible: http://www-ai.ijs.si/eliza-cgi-bin/eliza_script

JULIA: chatbot basado en Eliza, participó en el concurso Loebner del año 1994, salió cuarta de cinco chatbots.

disponible: http://www.lazytd.com/lti/julia/

MITBOLEL: Es un chatbot susceptible de ser adaptado y cambiado (*cuztomizable*)

disponible: http://www.romahi.com/yazann/Mitbolel/Mitbolel.html

THOUGHT TREASURE: En este chatbot el enfoque es diferente, ThoughtTreasure es una plataforma que comprende y procesa lenguaje natural, y que puede "razonar con sentido común", según su creador Erik T. Mueller.

disponible: http://www.signiform.com/tt/htm/tt.htm

BRIAN: Este chatbot está escrito en C++, extiende la idea general del "terapista" que utiliza ELIZA. Ganó el tercer lugar en el concurso Loebner 1998.

disponible: http://www.strout.net/info/science/ai/brian/

DR ABUSE: Un programa basado en ELIZA que responde en castellano.

disponible: http://dr-abuse.softonic.com/ (hay que descargarlo)

Eugene Goostman: Segundo en el concurso Loebener de 2005.

disponible: http://www.mangoost.com/bot/

Robin: Es un chatbot creado por el Ministerio de Sanidad y Consumo de España. Su función es informar a los jóvenes a través de Messenger sobre enfermedades de transmisión sexual y consumo de alcohol (utiliza una implementación de AIML en .NET)

disponible como contacto MSN: robin@msc.es

Encarta(R) Instant Answer: Otro chatbot para MSN, diseñado por Microsoft para promocionar Encarta (utiliza una implementación de AIML en .NET),
disponible como contacto MSN: encarta@botmetro.net

Si sobre la lista anterior se aplica el tercer requisito, quedan solo dos chatbots: ALICE, y Eliza. En ambos casos no solo se dispone de acceso al código fuente e información de su base de datos (o base de conocimientos) sino que además hay numerosa documentación sobre ellos, incluyendo la especificación en sus modo de funcionamiento ver [20] y [19]. Además el *engine* de ALICE, AIML, es utilizado en la construcción otros chatbots que tienen fines prácticos reales, como los últimos dos mencionados en la lista anterior.

1) Descripción de los sistemas de información existentes

En esta sección se hará un una descripción detallada de los sistemas existentes que se han tomado como casos de estudio. La forma escogida para mostrar su funcionamiento principal es la de un diagrama de flujo. De esta forma se podrá visualizar claramente similitudes y diferencias entre ambos chatbots y porque ALICE puede imitar el funcionamiento básico de ELIZA.

a) Diagramas de flujo de ALICE y ELIZA

A continuación se describirá de manera esquemática, mediante diagramas de flujo el funcionamiento de los dos chatbots antes seleccionados. Los gráficos anteriores describen el comportamiento general de los sistemas pero no dicen como evalúan el contexto, si es lo que hacen, en una conversación.

En primer lugar el contexto de cada palabra individual dentro de una frase o ambigüedad léxica se maneja de manera bastante satisfactoria en ambos sistemas ya que trabajan a nivel de oraciones completas (frases), la mínima unidad que pueden analizar como entrada es una frase. Incluso en el caso particular de que la frase de entrada esté compuesta por una sola palabra esta será tratada como una frase.

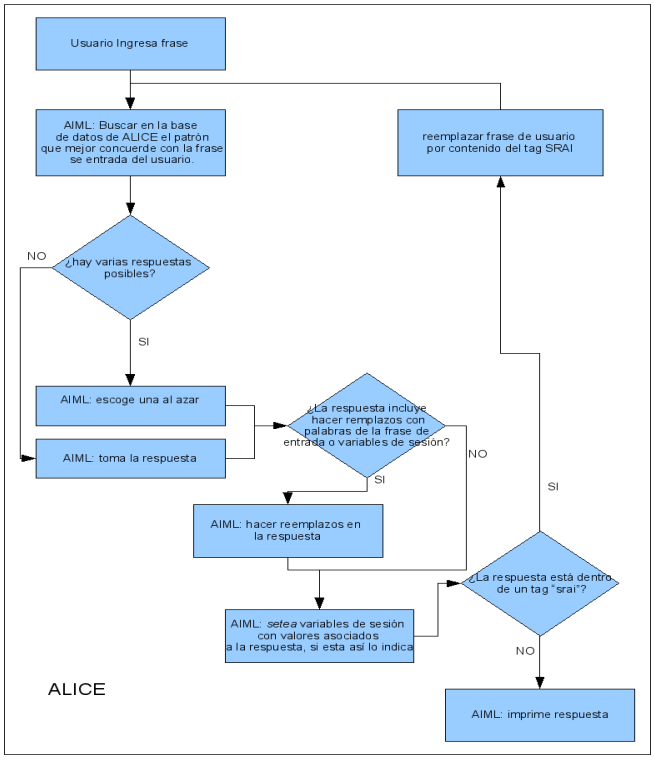

Fig. 12. Diagrama de flujo de ALICE

En segundo lugar el manejo de la ambigüedad sintáctica y la metonimia que son problemas de referencia interna dentro de una misma frase son manejados particularmente bien en ALICE ya que AIML mapea, generalmente, una frase entera a un patrón asociado a la respuesta, no descompone la frase o trata de analizarla.

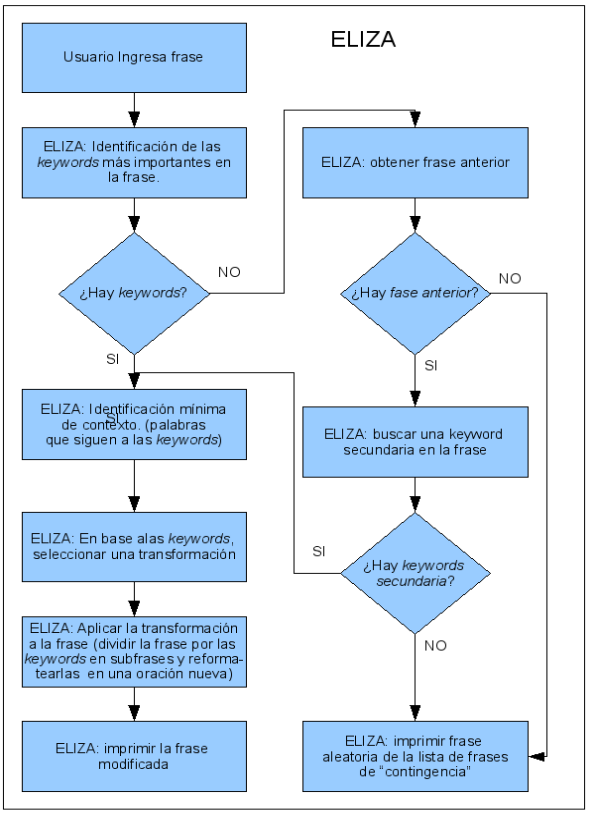

Fig. 13. Diagrama de flujo de ELIZA

Aunque puede darse el caso, cuando la frase coincide solo parcialmente con el patrón y este en vez de una respuesta tiene asociado una instrucción de recursividad (mediante el *tag srai*.) En relación a la pragmática, donde el problema reside en completar la información faltante de una frase con información del contexto, Eliza y ALICE funcionan de manera diferente. Ambas pueden dar respuestas diferentes a una misma frase de entrada en función de las frases anteriores intercambiadas entre el chatbot y el usuario.

En el caso de Eliza, cuando no tiene reglas asociadas o no encuentra palabras claves en una frase de entrada, uno de los procedimientos que puede ejecutar consiste en armar una respuesta, no sobre la frase actual, sino sobre alguna frase anterior del usuario distinta a la que se elaboró originalmente. De esta manera simula reencauzar o bien darle continuidad a una conversación como si en efecto la estuviese siguiendo aunque no es una verdadera respuesta en función del contexto ya que lo que logra es crear una conversación nueva.

En el caso de ALICE la funcionalidad para responder teniendo en cuenta el contexto está más y mejor elaborada. ALICE tiene dos maneras, que pueden actuar simultáneamente, para responder en base al contexto. Una es el uso de variables de estado. ALICE asocia a algunas respuestas variables de sesión. Cuando encuentra una respuesta dada, antes de imprimirla *setea* ciertos valores; por ejemplo, la respuesta a una frase de entrada que no tuvo ninguna coincidencia especifica podría ser "What's your favorite car?", en este caso ALICE saeteará la variable de sesión "topic" en "CARS". De este modo si para la próxima frase del usuario hay dos coincidencias iguales, ALICE escogerá la respuesta que esté dentro del tópico "CARS" (Las respuestas pueden estar dentro de "tópicos") [19]. La segunda manera en la cual ALICE puede responder en base al contexto es el *tag* especial de AIML "that", este *tag* está asociado a ciertas respuestas y lo que contiene es la frase textual que ALICE respondió anteriormente. El siguiente ejemplo, ilustrará mejor la función de este *tag:* Si ALICE a cierta frase de entrada de un usuario responde con "Do you like movies?" y el usuario responde a su vez a ello con la frase de entrada "yes". ALICE podría buscar en su base de datos, no solo un patrón que coincida con "yes", sino un patrón que coincida con "yes" y tenga asociado un *tag* "that" que coincida con "Do you like movies?".

Sí bien estas funcionalidades le permiten a ALICE manejar mejor el contexto, no bastan, como demuestran las transcripciones de sus conversaciones (o la de los otros *chatbots* basados en AIML.) El *tag* "that" exige que se duplique exponencialmente la base de datos de ALICE y solo permite contemplar la frase anterior a la actual.

La siguiente observación puede aclarar el porqué se siguen observado problemas en la evaluación del contexto en las conversaciones sostenidas por chatbots basados en AIML: si ALICE tiene en su base de datos de respuestas, hipotéticamente, 100 preguntas susceptibles de ser respondidas solo con "sí" (yes) o "no" (no) tendrá que tener 100 patrones "sí" y 100 patrones "no" asociados a cada pregunta y no se estarían contemplando los casos en los cuales el usuario decidió explayarse en la respuesta (por ejemplo "I like movies" en lugar de solo "yes" o "I don't like movies" en lugar de solo "no", solo para estos casos tendríamos 200 patrones más en la base.) (No habría problema con los sinónimos de "si" o "no", es decir si el usuario responde "afirmative" no sería necesario crear una nueva entrada, solo una tabla de sinónimos mediante el *tag* "srai")

C. Definición de requisitos del sistema

En el punto anterior se mostró una lista preliminar de requisitos de sistema que sirvió para entender mejor los puntos que siguieron y definir la situación actual.

A los requisitos anteriores habría que agregarle el siguiente: construcción de algún sistema auxiliar que permita discriminar la información en la base de datos del chatbot: esto es requerido para poder separar y analizar las frases y respuestas del sistema y distinguir los casos favorables, aquellos que deben seguir funcionando igual y los que deben ser modificados. Está tarea se puede descomponer en tres sub tareas que formarán parte de las tareas 2 a 4 descriptas anteriormente.

1) Estudio de alternativas de solución

A partir de este punto se presentan tres conjuntos de dos alternativas cada uno. Por un lado tomar a Eliza como punto de partida para el sistema que se pretende construir y por el otro lado utilizar a ALICE. Además están los subsistemas que se integrarán al chatbot: el sub-sistema detector y el sub-sistema conversor. Cada uno de estos sub-sistemas puede ser implementado de varias formas diferentes, en este estudio se evaluarán solo dos posibles formas: algorítmica y mediante una red neuronal.

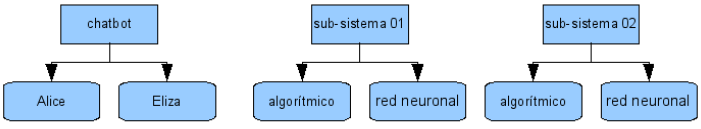

Fig. 14. Tres opciones

2) Chatbots

En primer lugar se analizará cual de los dos chatbots, se tomará como punto de partida, para ello se utilizará la siguiente tabla comparativa:

TABLA X. ALICE VS ELIZA (1)

ALICE
Intenta responder como lo haría una persona, su conversación no define un propósito o ámbito especifico.
Ha ganado tres veces el certamen "Loebner", ha sido finaliza otras dos.
ganó por última vez en el 2004
Se utiliza su *engine* para hacer chatbots con propósitos reales.
No posee grandes problemas de ambigüedad léxica, ambigüedad sintáctica y resolución de metonimia
Posee dos mecanismos diferentes para evaluar el contexto y emitir en base a ellos una respuesta.
AIML puede ser utilizado para construir un chatbot que simule el comportamiento de Eliza

TABLA XI. ALICE VS ELIZA (2)

Eliza
Intenta responder como lo haría un pisco terapeuta
Ganó el mismo certamen 4 veces y salió cuarta en otra oportunidad. (en verdad chatbots basados en Eliza, no Eliza propiamente dicha)
ganó por última vez en 1995
Finalidad meramente académica
No posee grandes problemas de ambigüedad léxica. Pero ambigüedad sintáctica y resolución de metonimia no están contemplados
no posee un verdadero mecanismo de evaluación del contexto (ver punto: 5.1.2.2)
Eliza (o su *engine*) no puede simular el comportamiento de ALICE

3) Sub-sistema detector

El sistema que se encargará de definir si una oración determinada es independiente del contexto o no, deberá poder hacer esta clasificación haciendo manejo de alguna técnica de procesamiento de lenguaje natural.

Desde el punto de vista algorítmico el tratamiento del lenguaje natural según Stuar Rusell y Peter Norving en [15] requiere de lo siguiente:

1. Definir un léxico, esto es una "lista de palabras permitidas. Las palabras [del léxico a su vez] se agrupan en categorías o partes del habla conocidas por los usuarios del diccionario: sustantivos, pronombres y nombres para denotar cosas, verbos para denotar sucesos, adjetivos para modificar sustantivos y adverbios para modificar verbos [...]" [15]
2. Una gramática para dicho léxico, "Una gramática es un conjunto finito de reglas que especifican un lenguaje." [15]. Como los lenguajes naturales al contrario que los lenguajes artificiales no tienen gramáticas formalmente definidas y la cantidad de reglas es muy grande hay que recurrir a la utilización de Gramáticas Aumentadas y otras herramientas como la subcategorización de verbos, etc.
3. Un algoritmo que a partir del léxico y la gramática realice un análisis sintáctico de una frase. Este algoritmo dependiendo de la precisión requerida, puede ser realizado de diversas maneras: ascendente, descendente, utilizando grafos, etc.
4. Finalmente sobre el árbol sintáctico que devuelve la función encargada de realizar el análisis sintáctico, se aplica un nuevo algoritmo encargado de realizar una interpretación semántica, esto es: la extracción del significado de la frase. Sin embargo, tal vez, no es necesario conocer el significado de una frase para poder decidir si es independiente o no del contexto. Del análisis semántico solo interesa una parte: la interpretación pragmática, la cual "[se encarga] del problema de la completitud de la información mediante la adición de información dependiente del contexto sobre la situación actual de cada interpretación candidata [que generó el análisis semántico]" [15]

A partir de este punto no hay una manera conocida y eficiente para realizar la interpretación pragmática, el casó más simple de resolver es el "significados de referentes", que son frases que hacen referencia directamente a la situación (contexto) actual, ejemplo: él, ello, allá, etc. Para estos casos se puede utilizar un algoritmo conocido, como el *pronoun references* de Hobbs [6] que logra una eficacia considerablemente alta aunque requiere de un análisis gramatical correcto o el *Pronominal Anaphora Resolution* de [7] que logra una eficacia similar o mayor en algunos casos.

Requisitos para un desarrollo basado en redes neuronales: en este caso lo que se necesita es una red que pueda recibir como parámetro de entrada una frase, o una serie de valores numéricos que representen a la frase y una salida booleana (suponiendo que no se utilizará lógica borrosa) que devuelva verdadero (ó 1) si la frase es independiente del contexto y falso (ó 0) si no lo es. Dentro de los dos grandes tipos de redes neuronales: auto-organizadas (SOM) y supervisadas habría que descartar a priori las redes SOM ya que en este escenario no buscamos agrupar datos por similitud o clasificar las frases en conjuntos inciertos sino en dos conjuntos bien definidos: frases independientes del contexto y frases dependientes del contexto.

Según Bonifacio Martín del Brío y Alfredo Sanz Molina en [12] un perceptrón multicapa "es capaz de representar complejos *mappings* y de abordar problemas de clasificación de gran envergadura, de una manera eficaz y relativamente simple." Además cuenta con el aval de haberse utilizado satisfactoriamente en muchos casos complejos y diversos como, por ejemplo, para determinar que bancos entrarían en quiebra durante la crisis bancaria española entre 1977 y 1985. Por otro lado un perceptrón multicapa (con una única capa oculta) es un aproximador universal de funciones como quedó demostrado con el teorema de Funahashi [3]. Estas consideraciones hacen al perceptrón una red neuronal artificial adecuada para emprender el proceso de construcción del sistema.

Como algoritmo de aprendizaje para esta red hay que considerar al más difundo y utilizado: *BackPropgation* el cual si bien presenta el inconveniente de que "busca minimizar la función de error pudiendo caer en un mínimo local o algún punto estacionario sin llegar a encontrar el mínimo global" [4]. No necesariamente un mínimo global es la única solución para

que el perceptrón proporcione un resultado exitoso como sucede en los sistemas de Codificación de la información, Traducción de texto en lenguaje hablado o Reconocimiento de caracteres en los cuales se utiliza de manera satisfactoria el algoritmo *BackPropagation.* [4]

4) Sub-sistema conversor

Para la implementación de este sistema ya se cuenta con algunos desarrollos bastante exitosos que podrían ser adaptados, por ejemplo el algoritmo de Hobbs de resolución por referencia, mencionado anteriormente. Sin embargo esto no cubre la totalidad del trabajo que debería realizar este subsistema ya que debe de poder encontrar información de contexto en casos más generales, como por ejemplo, si un usuario responde "yes" a la pregunta "Do you like movies?". En este caso debería de poder transformar "yes" en "I like movies" y aún más deberá de poder pedir información de contexto en casos en los cuales está no esté presente en el dialogo.

Si se piensa en una implementación para este sistema del tipo basado en redes neuronales, dicha red debería de poder determinar "que" es lo que la frase necesita para que esta sea independiente del contexto y luego tomando el dialogo como entrada de la red extraer dicha información, con lo cual se requerirá convertir todo el dialogo entre el usuario y el chatbot en una serie de valores de entrada para la red y luego *mapear* el resultado de salida a alguna porción de frase del dialogo. Con lo cual es de esperar que dicha red tuviera un grado considerable de complejidad.

D. Valoración de alternativas

La tabla *XI* muestra claramente que ALICE es un sistema más moderno y por lo tanto más complejo y que ha dado mejores resultados que Eliza en el área de los chatbots inteligentes, también es posible ver que incorpora rudimentos de detección de información contextual (interpretación pragmática). Todo esto posiciona a ALICE y a su lenguaje de construcción: AIML por encima de Eliza.

El problema de identificación de independencia del contexto en una frase es un problema que involucra las siguientes características:

- No hay una regla clara que pueda usarse a priori para diferenciar estas dos clases de frases.
- Las frases que pueden ser clasificables son en principio infinitas.
- Es un problema relativamente sencillo para un ser humano.
- La regla si es que existe hay que extraerla de un conjunto de ejemplos.
- Los métodos algorítmicos implican el desarrollo de un analizador sintáctico (y quizás semántico) considerablemente preciso.

Si bien los ítems anteriores son bastante evidentes tampoco terminan de definir la mejor solución, sin embargo por la naturaleza del problema apuntan más hacia una solución de tipo no algorítmica. Por lo cual, según los casos analizados, la utilización de una red neuronal será la solución más propicia.

El sub-sistema conversor presenta características clásicas de sistemas algorítmicos como búsqueda de palabras (referentes) y reemplazo de sub cadenas. Por ello y por el hecho de que existen sistemas algoritmos que podrían ser utilizados como referencia o puntos de partida la solución adoptada para este subsistema será finalmente algorítmica.

E. Selección de la solución

En este punto ya se ha escogido una solución y una manera de encarar el desarrollo de la investigación, se ha obtenido una planificación tentativa y un detalle de los módulos que contendrá y de los sistemas que servirán como base de la misma. Esta información se resume en la siguiente lista de módulos:

- Modulo ALICE original
  - importación de todo su base de datos a una base de datos relacional
- Sub-sistema detector (red neuronal)
  - sub-sistema para convertir frases en entrada de la red neuronal.
- Sub-sistema conversor (algorítmico)

VI. Conclusiones

Para el problema de la clasificación de frases en los grupos "dependiente del contexto" e "independiente del contexto", no se encontró una solución óptima, sin embargo se encontró una solución suficientemente buena como para completar el desarrollo de la Investigación. El perceptrón multicapa con *Back Propagation* que utilizó como *input* el primer mapa generado compuesto de un vector de números en punto flotante de 15 posiciones logró un 54 % de efectividad mientras que el perceptrón multicapa con *Back Propagation* que utilizó el segundo *input,* la matriz de 15x5 números en punto flotante logró una efectividad de solo el 36 %.

Por su parte la red SOM, Kohonen no logró construir conjuntos de frases de forma tal que la red *Back Propagation* aplicada a cada uno de los subconjuntos generase un error cuadrático medio por debajo del error cuadrático medio hallado para el conjunto total. En cada caso existió un conjunto con un error cuadrático medio igual o superior al que la red generaba (tras un mismo número de iteraciones) sobre el conjunto general.

Es difícil ignorar la incompletitud de la base de datos "Lexico" utilizada para generar los mapas y la posibilidad de saber hasta qué punto los mapas creados representan las propiedades de las frases que interesa analizar. Sin duda ambos pueden ser mejorados, la dificultad radica en saber cómo generar mejores mapas de entrada y en como optimizar la clasificación de las palabras para aumentar los casos positivos.

Si bien el método normal de resolver problemáticas relacionadas con la sintaxis de una sentencia es a través de algoritmos deterministicos, la implementación de esta solución a través de redes neuronales tuvo un resultado satisfactorio.

El algoritmo de resolución de referentes: *RAP,* si bien no logró una tasa tan alta de efectividad como la descripta en [7] logró una efectividad suficientemente buena como para que el chatbot logre responder de forma satisfactoria a muchas de las frases con referencias anafóricas (que anteriormente no lograba responder correctamente). La disminución en la efectividad se debió a dos factores: el cambio del parser de procesamiento de lenguaje natural, ya que no se consiguió el mismo utilizado y fue reemplazado por otro de la biblioteca "Open-NLP" (que difiere levemente en su forma de comportarse.) La segunda razón es el cambio radical del tipo de discurso empleado, en [7] midieron la eficacia utilizando textos de manuales de computadoras y no diálogos coloquiales casuales.

Por último el procedimiento encargado de reformular las respuestas a preguntas cerradas logró un éxito considerable, mayor del esperado ya que de una forma muy simple logra transformar las preguntas del chatbot en sentencias declarativas. Es un cambio que podría implementarse de forma productiva tal y como está de forma independiente de las otras mejoras.

VII. Futuras líneas de investigación

Dentro de la misma problemática de la interpretación pragmática en chatbots quedan varios aspectos para seguir investigando y mejorando la calidad de las respuestas. Por un lado se tiene el problema de la clasificación de las frases en "independientes del contexto" y "dependientes del contexto", dicha problemática podría ser mejorada, en principio, de dos formas:

Mejorando los sistemas inteligentes propuestos en esta investigación para así obtener una tasa de efectividad más elevada al clasificar los casos, lo cual podría lograrse, por ejemplo, aplicando lógica borrosa y sistemas difusos para todos aquellos casos que no pueden ser definidos de forma univoca dentro de uno u otro conjunto.

La otra forma de mejorar la clasificación sería utilizando métodos algorítmicos deterministicos, sería interesante medir la tasa de efectividad de estos en la clasificación de casos contra una clasificación hecha con sistemas inteligentes. Una forma algorítmica de encarar el problema podría ser a partir del algoritmo *RAP*, modificándolo para que cumpla este objetivo.

Por otro lado se podría aumentar la tasa de éxito del algoritmo *RAP*, mejorarla adaptando el algoritmo al tipo de discurso propio de los diálogos casuales sería una opción interesante como trabajo de investigación. También se podría incorporar un sistema para que el algoritmo "aprenda" de los diálogos; en el sentido de incorporar nuevas palabras y nuevos casos de conocimiento del tipo "respuesta-estimulo" como es el caso de AIML, también nuevos sustantivos, en especial nombres propios para el caso de *RAP, y*a que los nombres propios son sustantivos muy comunes y difícilmente estén todos en una base de datos armada previamente.

Nuevas líneas de investigación se abren al encarar otros problemas devenidos también por la falta de interpretación pragmática en los chatbots al generar sus respuestas. Estos problemas fueron descriptos en el punto 3, son la ambigüedad léxica en una sentencia, la elipsis, etc.

Referencias

[1] CIRG. 2008. *Cybernetic Intelligence Research (Group) – Loebner Prize 2008*. http://www.rdg.ac.uk/cirg/loebner/cirg-loebner-main.asp, página vigente al 31/08/08

[2] Fitrianie, S. 2002, *My_Eliza A multimodal Communication System*, Master of Science thesis, Delft University of Technology. http://www.kbs.twi.tudelft.nl/Publications/MSc/2002-Fitriani-MSc.html. página vigente al 31/08/08

[3] Funahashi, K. I. 1989 *On the approximate realization of continuous mappings by neural networks*. Neural Networks, 2, pag. 183-192

[4] Ramón García Martínez, Magdalena Servente, Diego Pasquini 2003. *Sistemas Inteligentes*. Editorial Nueva Libreria.

[5] Hilera J. R., Martínez V.J. 1995. *Redes Neuronales Artificiales: Fundamentos, Modelos y Aplicaciones*. Editorial RA-MA

[6] Hobbs, Jerry 1978.*Resolving pronoun references,* Lingua 44, pag. 311-338.

[7] Lappin, S. and Leass, H.J. (1994). *An algorithm for pronominal anaphora resolution*. *Computational Linguistics* 20(4),pag. 535-561

[8] Loebner Contest, 2006. *Loebener Prize 2006 Information,* http://loebner.net/Prizef/2006_Contest/loebner-prize-2006.html, página vigente al 04/04/09

[9] Loebner Contest, 2007. *17th Annual Loebner Prize for Artificial Intelligence,* http://loebner.net/Prizef/2007_Contest/loebner-prize-2007.html, página vigente al 04/04/09

[10] Loebner Contest, 2008. *Loebener Prize 2008 Information, http://loebner.net/Prizef/2008_Contest/loebner-prize-2008.html,* página vigente al 04/04/09

[11] Loebner Contest, 2009. *Loebener Prize 2009 Information, http://www.loebner.net/Prizef/loebner-prize.html, página vigente al 04/04/09*

[12] Bonifacio Martín del Brío, Alfredo Sansz Molina 2001. *Redes Neuronales y Sistemas Difusos.* Editorial RA-MA

[13] Müller, B., Reinhardt, J. *Neural Networks*. *An Introduction*, Springer-Verlag, 1990

[14] Penrose, Roger 1989. *The Empreror's New Mind.* Oxford Univesity Press.

[15] Stuart Rusell, Peter Norving 2003. *Inteligencia Artificial, Un enfoque Moderno.* Editorial Prentice Hall

[16] Searle, John 1980. *Mind, brains and programs.* Behavioral and Brain, Sciences. vol 3, nº 3, pag. 417-457

[17] SOM_PACK. *The self-Organizing Map Program Package.* Helsinki University of Technology, Finland, abril 1995.

[18] Turing, A. 1950. *Computing Machinery and Intelligence*. Mind. Vol. LIX, N° 236, pag. 433-460

[19] Wallace, R. 2003. *The Elements of AIML Style*. ALICE A.I. Foundation.

[20] Weizenbaum, J. 1966. *ELIZA – A computer Program for the Study of natural Language Communication Between Man And* Machine, Communication of the ACM. Vol 9, N° 1, pag. 36-45

[21] Loebner Contest, 2004. Loebener Prize 2004 Information: *http://loebner.net/Prizef/2004_Contest/Wallace.html* página vigente al 04/04/09

Juan Manuel Rodríguez. Es Ingeniero en Informática en la Universidad de Buenos Aires en el año 2012. Actualmente es docente de las materias: Sistemas de Soporte para Celdas Producción Flexible y Sistemas de Programación no convencional de Robots. Su campo de interés en investigación es el procesamiento del lenguaje natural.

Hernán Merlino. Es Magister en Ingeniería de Software por la Universidad Politécnica de Madrid. Es Profesor Titular de la Licenciatura en Sistemas y Director del Laboratorio de Investigación t Desarrollo en Arquitecturas Complejas de la Universidad Nacional de Lanús. Es Docente Investigador del Programa de Incentivos de la Secretaria de Políticas Universitarias del Ministerio de Educación. Su área de investigación es patrones de diseño y de desarrollo de software.

Enrique Fernández. Es Doctor en Ciencias Informáticas por la Universidad Nacional de La Plata. Es Investigador Adscripto al Grupo de Investigación en Sistemas de Información de la Licenciatura en Sistemas de la Universidad Nacional de Lanús. Es Docente Investigador del Ministerio de Educación de la Republica Argentina.