PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA

MÓDULOS DE ATENCIÓN TEMPORAL PARA REDES NEURONALES CON MEMORIA EXTERNA

RODOLFO PALMA OTERO

Tesis para optar al grado de
Magíster en Ciencias de la Ingeniería

Profesor Supervisor:
ÁLVARO SOTO ARRIAZA

Santiago de Chile, Junio 2020

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA

MÓDULOS DE ATENCIÓN TEMPORAL PARA REDES NEURONALES CON MEMORIA EXTERNA

RODOLFO PALMA OTERO

Miembros del Comité:
ÁLVARO SOTO ARRIAZA
JORGE BAIER ARANDA
LUIS MARTÍ OROSA
WERNHER BREVIS VERGARA

Tesis para optar al grado de
Magíster en Ciencias de la Ingeniería

Santiago de Chile, Junio 2020

A mis padres y hermano

AGRADECIMIENTOS

A continuación unas sinceras palabras de agradecimiento a quienes me apoyaron durante el desarrollo de este trabajo. En primer lugar, agradezco el amor y apoyo constante de mis padres, Marcela y Rodolfo, y mi hermano, Juan Antonio. A lo largo de los altos y bajos inherentes de cualquier investigación, me aconsejaron y entregaron todo lo que necesité para realizar esta tesis.

Luego, agradezco a mi supervisor, Álvaro Soto. En julio de 2017 le envié un correo electrónico sin que él me conociera, expresando mi interés en investigar temas de aprendizaje de máquina. Álvaro me abrió todas las puertas del entonces GRIMA, hoy IA Lab del DCC de la Facultad, donde conocí grandes personas unidas por el interés común de empujar la frontera del conocimiento en inteligencia artificial, sin perder la perspectiva del impacto que estas tecnologías pueden causar en la sociedad.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ABSTRACT

This work introduces two temporal attention modules, which can be plugged into traditional memory augmented recurrent neural networks in order to improve their performance in natural language processing tasks. The temporal attention modules provide new inductive biases to the models allowing them to compute attention distributions over the different time steps of the input sequence. The values of these attention distributions can be inspected in order to identify the sequence's elements that the model considered relevant during the inference.

Using the Entity Network (Henaff, Weston, Szlam, Bordes, & LeCun, 2016) as the model backbone, experiments were made on a set of question answering tasks called bAbI tasks. Due to the addition of the temporal attention modules, the performance metric increased 26% when the temporal attention was supervised, and 13,5% when it wasn't. Moreover, the usage of temporal attention modules proved useful at resolving reasoning tasks that the original model was unable to solve.

Keywords: artificial intelligence, machine learning, deep learning, natural language processing.

RESUMEN

En este trabajo se introducen dos módulos de atención temporal, los cuáles pueden ser acoplados a modelos tradicionales de redes neuronales recurrentes con memoria externa para mejorar el rendimiento de éstos en tareas de procesamiento de lenguaje natural. Los módulos de atención temporal otorgan a los modelos la capacidad de construir una distribución de atención sobre los diferentes instantes de tiempo de la secuencia de entrada, permitiendo posteriormente observar cuáles fueron los elementos considerados relevantes para realizar la inferencia. Esta capacidad agrega atributos de explicabilidad a los modelos.

Usando como base la *Entity Network* (Henaff et al., 2016), se realizaron experimentos en tareas de *question answering* utilizando el conjunto de datos bAbI tasks. Gracias al uso de los módulos de atención temporal se incrementa la métrica de rendimiento en un 26% al supervisar la atención temporal y en un 13,5% al no hacerlo. Adicionalmente, el uso de de los módulos de atención temporal permite resolver tareas de razonamiento que el modelo original no es capaz de resolver.

Palabras Claves: inteligencia artificial, aprendizaje de máquina, aprendizaje profundo, procesamiento de lenguaje natural.

1. INTRODUCCIÓN

1.1. Motivación

El aprendizaje profundo, o *deep learning*, es un campo del aprendizaje de máquina que durante la última decada ha impactado diversos aspectos de la sociedad (LeCun, Bengio, & Hinton, 2015; Brynjolfsson & Mitchell, 2017). Pese al desarrollo que ha experimentado este campo de la inteligencia artificial, existe cierto grado de consenso de que queda camino por recorrer para desarrollar una inteligencia artificial general, capaz de emular por completo o incluso superar el funcionamiento neurológico del ser humano (Marcus, 2018).

Los pilares fundamentales del funcionamiento del aprendizaje profundo son el aprendizaje de características jerárquicas composicionales y el procesamiento de datos con múltiples etapas de cómputo. El primer pilar permite que múltiples capas de procesamiento aprendan representaciones de información con diferentes niveles de abstracción. Por otra parte, el segundo pilar otorga al usuario herramientas para inducir sesgos que emulen el razonamiento necesario para resolver alguna tarea.

En general, el aprendizaje profundo funciona gracias a la conjugación de tres elementos: 1) un espacio de hipótesis dado por la estructura paramétrica del modelo, 2) una función de pérdida que indica cuán bien se encuentra ajustado el modelo a los datos de entrenamiento, y 3) un método para ajustar los parámetros del modelo a fin de minimizar la función de pérdida. Las decisiones que toma el implementador de un modelo en cualquiera de esos tres aspectos induce sesgos en el funcionamiento del algoritmo.

Este tipo de sesgos pemite generalizar a ejemplos más allá de los observados durante la etapa de entrenamiento (Mitchell, 1980). Existe consenso de que es importante continuar introduciendo sesgos inductivos en modelos profundos con el objetivo de facilitar el aprendizaje de entidades, relaciones, y reglas de composición de estos elementos (Battaglia et al., 2018).

Los progresos recientes en arquitecturas profundas raramente consideran como un atributo de primera categoría a la explicabilidad (Freitas, 2014; Guidotti et al., 2018). En general, se privilegian modelos que mejoran los estados del arte sin cuestionar la característica de "caja negra" que éstos tienen. Una causa posible de este comportamiento es la percepción de la existencia de una dicotomía entre explicabilidad y capacidad de aprendizaje: árboles de decisión son modelos fácilmente interpretables por su naturaleza gráfica, pero tienen baja capacidad de aprendizaje comparados con modelos profundos, que a su vez tienen pobres o inexistentes atributos de explicabilidad.

Un debate que se ha producido en la comunidad de inteligencia artificial es sobre la importancia de agregar atributos de explicabilidad a los modelos de aprendizaje profundo. El principal argumento de los detractores de esta idea es que el usuario final está principalmente preocupado de obtener buen rendimiento en la tarea particular y no en obtener buenas explicaciones. Por otra parte, los miembros de la comunidad a favor fundamentan que es importante agregar atributos de explicabilidad, a fin de convencer a aquellos actores de la sociedad con aprensiones de aplicar modelos profundos en procesos complejos donde es crítico justificar las inferencias realizadas.

En esta tesis se busca romper esta dicotomía mediante la agregación de atributos de explicabilidad a modelos profundos con comportamiento de "caja negra", sin sacrificar rendimiento en la tarea que se está resolviendo. En particular, usando como modelo base una red neuronal recurrente con memoria externa, denominada *Entity Network* (Henaff et al., 2016), aplicada a la tarea de procesamiento de lenguaje natural de *question answering*, se diseñarán dos módulos que implementen sesgos inductivos con nociones de atención (Bahdanau, Cho, & Bengio, 2014) capaces de incrementar el rendimiento del modelo, al mismo tiempo que agreguen atributos de interpretabilidad ausentes en el modelo original.

Los módulos diseñados serán evaluados en la tarea de procesamiento de lenguaje natural conocida como *question answering*. En esta tarea, el modelo recibe una secuencia de hechos en lenguaje natural. A continuación, recibe una consulta, también en lenguaje natural, respecto a la información recibida en la secuencia anterior. Finalmente, infiere una

1 Wolves are afraid of mice.
2 Sheep are afraid of mice.
3 Winona is a sheep.
4 Mice are afraid of cats.
5 Cats are afraid of wolves.
6 Jessica is a mouse.
7 Emily is a cat.
8 Gertrude is a wolf.
Consulta: What is Emily afraid of?
Respuesta: wolf

Figura 1.1. Ejemplo de entrenamiento de tarea de *question answering*. Se observan la secuencia de hechos en lenguaje natural, una consulta respecto a esta información y la respuesta a la consulta. Para responder correctamente este ejemplo, el modelo debe ser capaz de razonar de forma deductiva. Se destacan los hechos relevantes a la consulta.

respuesta a la consulta recibida en base a la secuencia de hechos procesada. La inferencia se realiza mediante una clasificación sobre clases asociadas a las posibles respuestas, que se asume son conocidas con antelación. Un ejemplo de entrenamiento de esta tarea se observa en la figura 1.1.

El trabajo expuesto en esta tesis introduce dos familias de módulos de atención temporal que pueden ser usados para agregar atributos de explicabilidad a las inferencias de modelos con memoria externa, mejorando su rendimiento en las tareas principales. Los mecanismos de atención temporal permiten que el modelo aprenda a identificar cuáles son los elementos de la secuencia de hechos que son relevantes para responder una consulta sobre la historia.

En esencia, la atención temporal permite al modelo inferir respuestas más fácilmente mediante la eliminación de ruido asociado a elementos de la secuencia de hechos que son irrelevantes para la consulta. Además, agrega atributos de interpretabilidad al modelo. La interpretabilidad se logra observando los puntajes de la atención temporal: elementos de la secuencia de hechos con puntaje alto fueron relevantes para responder la consulta, mientras que aquellos con puntajes pequeños no fueron considerados de esa forma por el modelo.

Se diseñarán e implementarán dos familias de módulos de atención temporal. La primera clase consiste en módulos *pre hoc*, es decir, que calculan la distribución de atención temporal antes de realizar el paso de inferencia de respuesta. Por otra parte, la segunda familia de módulos es del tipo *post hoc*, es decir, ejecutan *a posteriori* el cálculo de la atención temporal luego de haber realizado la inferencia. Ambas clases de módulos presentan ventajas y desventajas que serán analizadas.

El módulo *pre hoc* tiene la ventaja de agregar atributos de interpretabilidad que inciden directamente en la inferencia de la respuesta. Esto se debe a que los valores de la atención temporal calculada por este módulo son usados en las operaciones del modelo responsables de realizar la inferencia de la respuesta. Sin embargo, comparado al módulo *post hoc*, tiene la desventaja de no poder usar la respuesta inferida como información adicional en el cálculo de la atención temporal. El módulo *post hoc* sí cuenta con la respuesta inferida, por lo que tiene mayor información al momento de calcular la distribución de atención temporal.

Dado lo anterior, es esperable que el módulo *post hoc* tenga mejor rendimiento que el módulo *pre hoc* en la tarea de identificar los hechos relevantes de una historia. Pese a esto, el módulo *pre hoc* entrega una explicación más transparente dado que expone valores que son utilizados directamente en el paso de inferencia. Los valores expuestos por el módulo *post hoc* no necesariamente reflejan mecanismos de funcionamiento internos del modelo usados para la inferencia de la respuesta.

1.2. Hipótesis y objetivos

La hipótesis del trabajo expuesto en esta tesis es que la incorporación de módulos de atención temporal a redes recurrentes con memoria externa permiten incrementar el rendimiento de éstas en tareas de procesamiento de lenguaje natural, además de agregar atributos de explicabilidad inexistentes en los modelos originales.

En concordancia con la hipótesis propuesta, los objetivos específicos del trabajo son los siguientes: 1) implementar y replicar los resultados originales del modelo base, 2) diseñar e implementar un módulo de atención temporal post hoc, 3) diseñar e implementar un módulo de atención temporal pre hoc, y 4) evaluar y analizar el desempeño de ambos módulos respecto al modelo original.

2. REVISIÓN BIBLIOGRÁFICA

2.1. Aprendizaje profundo

Las técnicas de aprendizaje jerárquico composicional conocidas como *deep learning*, o aprendizaje profundo, consisten en la utilización de modelos con múltiples capas o pasos de procesamiento que aprenden a representar información con distintos niveles de abstracción (LeCun et al., 2015). Los paradigmas clásicos de aprendizaje de máquina, anteriores al *deep learning*, dependen fuertemente en la capacidad del implementador de seleccionar correctamente las características, o *features*, relevantes para los modelos, una práctica comúnmente denominada *feature engineering*. En el caso del aprendizaje profundo, se delega al modelo la responsabilidad de aprender las características importantes.

En Sutskever, Vinyals, y Le (2014) se propone la arquitectura *sequence–to–sequence* para resolver de forma general tareas que puedan ser expresadas como un mapeo entre una secuencia de origen y otra secuencia objetivo. Fundamentalmente, se codifica mediante una *Long Short–Term Memory* (Hochreiter & Schmidhuber, 1997), o LSTM, la secuencia de entrada en un vector de dimensionalidad fija, para luego ser decodificado por otra LSTM produciendo la secuencia objetivo. Muchas tareas de procesamiento de lenguaje natural pueden ser ajustadas a este *framework*: traducción entre idiomas, redacción de resúmenes, *question answering*, entre otras.

Una limitación del trabajo presentado en Sutskever et al. (2014) es que el vector intermedio de tamaño fijo constituye un cuello de botella de información en el caso de tareas complejas con secuencias de origen largas. En el trabajo de Bahdanau et al. (2014) se introducen mecanismos de atención que no usan vectores intermedios de dimensionalidad fija, sino que permiten que el modelo realice de forma automática búsquedas sobre representaciones relevantes presentes en la secuencia de origen.

Mecanismos de atención, como los introducidos en Bahdanau et al. (2014), pueden ser generalizados como distribuciones de atención sobre valores V a partir de llaves K y

consultas Q. Esencialmente, si se desea resumir un conjunto de valores (K_i, V_i) es posible calcular para cada i un puntaje de relevancia a través de mediciones de similitud entre las llaves K_i y la consulta Q. Luego, los puntajes calculados para los valores V permiten hacer una mezcla de sus valores formando vectores que resumen la información presente en éstos, poniendo especial énfasis en aquellos relevantes para responder la consulta Q. Gracias a este paradigma se alcanzaron nuevos estados del arte en tareas tales como traducción entre idiomas (Bahdanau et al., 2014) y descripción de imágenes (Vinyals, Toshev, Bengio, & Erhan, 2014), entre otras.

Una serie de modelos propuestos recientemente, denominados *Pointer Networks* (Vinyals, Fortunato, & Jaitly, 2015), han hecho uso de las nociones de atención previamente expuestas para un fin distinto. A diferencia de Bahdanau et al. (2014), donde la atención es usada para construir una mezcla de los estados ocultos de la secuencia de origen, en Vinyals et al. (2015) la atención es usada para apuntar directamente hacia la secuencia de origen, permitiendo realizar inferencias en tareas donde el vocabulario de respuestas no es conocido *a priori* mediante un enfoque extractivo.

En el trabajo desarrollado en esta tesis se utilizarán los mecanismos de atención mostrados anteriormente en sus dos modalidades: tanto para realizar mezclas del contenido de las memorias externas a lo largo del tiempo, así como también para apuntar a los elementos de la secuencia de entrada que más condicionan la inferencia del modelo.

2.2. Redes con memoria externa

De forma teórica se ha demostrado que las redes neuronales recurrentes (RNNs) son Turing completas (Siegelmann & Sontag, 1995), es decir, tienen el potencial de aprender cualquier algoritmo computacional, pero en la práctica ésta no es una capacidad sencilla de alcanzar. Un enfoque adoptado en los últimos años para facilitar el aprendizaje de modelos recurrentes ha sido el enriquecimiento de éstos mediante el uso de memorias

externas (Graves, Wayne, & Danihelka, 2014; Graves et al., 2016; Weston, Chopra, & Bordes, 2015; Sukhbaatar, Szlam, Weston, & Fergus, 2015).

Las memorias externas consisten en un conjunto de vectores que suelen ser accedidos mediante mecanismos de direccionamiento basados en localización o contenido. En esencia, esta familia de modelos se inspira en arquitecturas clásicas de computadores como la de Von Neumann (Von Neumann, 1993), donde además de registros de memoria presentes en las unidades de procesamiento y control, existen unidades de memoria externa que almacenan datos e instrucciones a largo plazo, o la máquina de Turing, donde se enriquece un autómata finito con una cinta de memoria infinita.

Un ejemplo de una red neuronal de memoria externa es la *Entity Network* (Henaff et al., 2016). La *Entity Network* se caracteriza por contar primero con un módulo de entrada, luego con un módulo de memorias externas que se actualizan en paralelo a medida que el modelo recibe nueva información, y finalmente con un módulo de salida. Las memorias se componen por una tupla de vectores (w, h) que corresponden a representaciones de una entidad e información de esa entidad, respectivamente. De esta manera, el modelo luego de identificar las entidades relevantes del mundo en el que se desenvuelve, es capaz de mantener un registro del estado de éstas.

Debido al planteamiento modular de la *Entity Network*, además del valioso contenido semántico que aprende en los vectores (w, h) de la memoria externa, ésta ha sido utilizada como un buen punto de partida para diseñar nuevos módulos que agreguen atributos y capacidades que el modelo original no posee. Por ejemplo, en Bansal, Neelakantan, y McCallum (2017) modificaron el módulo de memoria dinámica para incorporar capacidad de razonamiento relacional entre las entidades. El trabajo presentado en esta tesis sigue un enfoque análogo, modificando el módulo de salida para agregar nociones de atención temporal que permitan adquirir atributos de explicabilidad y mejorar el rendimiento del modelo original.

2.3. Question answering

Diseñar e implementar modelos que puedan responder consultas realizadas en texto natural, o *question answering*, ha sido una tarea clásica en la comunidad de procesamiento de lenguaje natural (NLP, por su sigla en inglés). Sin ir más lejos, la resolución de este tipo de tareas por parte de sistemas inteligentes ha sido un atributo deseable desde los inicios de la inteligencia artifical (Turing, 1950). Al igual que tareas relacionadas con visión por computador, el aprendizaje profundo ha provocado avances significativos en tareas de NLP (Hirschberg & Manning, 2015), principalmente debido a la capacidad que tiene el paradigma profundo de aprender representaciones distribuidas con múltiples componentes activas para cada palabra.

Diversos conjuntos de datos han sido propuestos para entrenar y evaluar modelos que trabajan sobre la tarea de *question answering*. En Y. Yang, Yih, y Meek (2015) proponen WikiQA, que consiste en pares de oraciones y preguntas, con sus respectivas respuestas, extraídos desde la Wikipedia en inglés. En Rajpurkar, Jia, y Liang (2018) se publica el conjunto de datos SQuAD, similar a WikiQA pero con mayor número de instancias de entrenamiento. La última versión de SQuAD incluye más de 150.000 ejemplos de entrenamiento, además de incorporar preguntas cuya respuesta no existe. Cada ejemplo de entrenamiento en SQuAD consiste en un párrafo de la Wikipedia en inglés, una pregunta sobre este párrafo y las posibles respuestas a la consulta.

La resolución de los conjuntos de datos WikiQA y SQuAD han concentrado gran parte de los esfuerzos de la comunidad de NLP en los últimos años (Devlin, Chang, Lee, & Toutanova, 2018; Z. Yang et al., 2019). No obstante, para los objetivos del trabajo presentado en esta tesis, ambos conjuntos de datos presentan dos dificultades. En primer lugar, no es posible analizar de forma clara las distintas capacidades de razonamiento que los modelos evaluados muestran, ya que solo se puede determinar de forma general la correctitud de las respuestas inferidas por el modelo. Finalmente, un objetivo importante de este trabajo es agregar la capacidad de apuntar hacia la información relevante que el

modelo considera a la hora de realizar inferencias, a fin de articular una explicación que el usuario final utilice para entender el comportamiento del modelo. Los conjuntos de datos expuestos anteriormente carecen de información respecto a la información relevante en los párrafos de contexto, por lo que resulta difícil medir el grado de logro de este objetivo.

Un conjunto de datos que resulta interesante bajo el punto de vista anterior son las *bAbI tasks* (Weston et al., 2016). Consisten en veinte tareas de *question answering* que permiten poner a prueba distintas habilidades de razonamiento cognitivo. Cada tarea consiste en 1.000 ejemplos de entrenamiento y 1.000 ejemplos de *test*. Cada instancia del conjunto de datos se compone de: una secuencia de hechos, una pregunta, una respuesta y los hechos relevantes, que consisten en el subconjunto de la secuencia de hechos con la información mínima necesaria para responder la consulta. La construcción de las tareas se realiza de forma sintética, mediante la simulación de un mundo donde distintas entidades interactúan entre si.

El trabajo presentado en esta tesis será evaluado sobre las *bAbI tasks*, debido a que permiten evaluar de forma precisa veinte diferentes habilidades de razonamiento. Además, cuentan con información respecto a los hechos relevantes de cada ejemplo de entrenamiento, la que podrá ser usada tanto para evaluar la capacidad explicativa del modelo, así como también supervisarlo mediante la incorporación de un término adicional a la función de pérdida de éste.

3. MÉTODO PROPUESTO

3.1. Entity Network

La columna vertebral del modelo presentado en esta tesis es una red neuronal denominada *Entity Network* (Henaff et al., 2016). La *Entity Network* es un modelo recurrente con memoria externa capaz de resolver tareas de *question answering* y diálogo, entre otras. Este modelo introduce sesgos inductivos con el objetivo de aprender a identificar entidades en el dominio de entrada, a fin de realizar seguimiento de propiedades relevantes para resolver alguna tarea de interés.

El modelo se descompone en tres módulos: uno de entrada, uno de memoria dinámica y finalmente uno de salida. Esta tesis propone módulos que extienden el módulo de salida, agregando atributos de explicabilidad. Los módulos originales serán descritos a continuación para contextualizar al lector.

3.1.1. Módulo de entrada

Como se explicó anteriormente, el trabajo presentado en esta tesis será aplicado a la tarea de *question answering*. Bajo esta configuración, el módulo de entrada de la *Entity Network* (Henaff et al., 2016) en el tiempo t recibe una secuencia de vectores $\{e_1, \ldots, e_L\}$, donde $e_i \in \mathbb{R}^e$ corresponde al *embedding* con dimensionalidad e de la i-ésima palabra perteneciente a la oración del instante de tiempo t que tiene largo L. El módulo de entrada está encargado de integrar esta información en un vector de dimensionalidad fija s_t para cada instante de tiempo t. Este vector s_t se calcula de la siguiente forma:

$$s_t = \sum_i f_i \odot e_i \qquad\qquad s_t, f_i, e_i \in \mathbb{R}^e \tag{3.1}$$

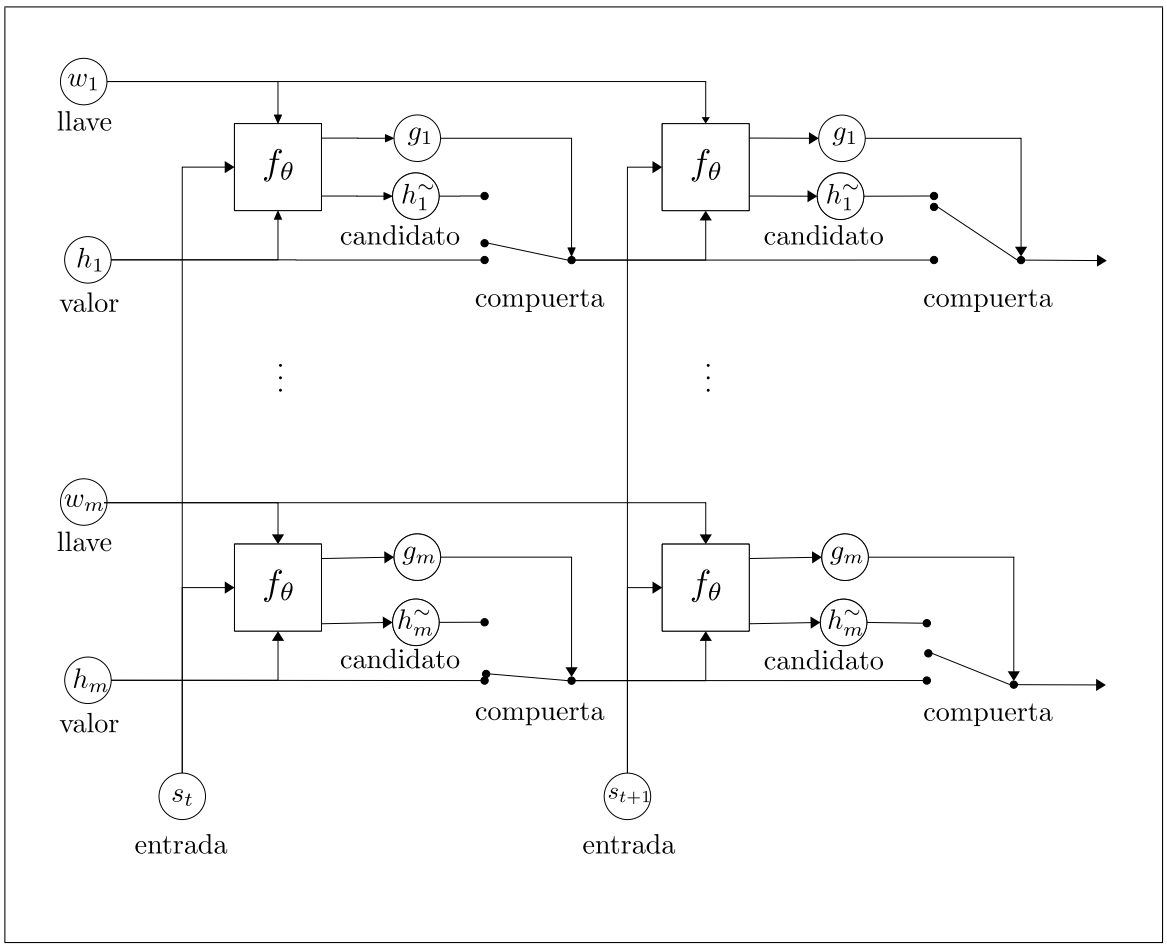

Figura 3.1. Se ilustra el funcionamiento del módulo de memoria dinámica del modelo de Henaff et al. (2016). Es posible observar la naturaleza recurrente del modelo, inspirado en redes con mecanismos de compuerta.

Las máscaras multiplicativas $\{f_1, \ldots, f_L\}$ forman parte del conjunto de pesos entrenables del modelo. Es posible observar que esta codificación corresponde a una generalización del método de representación *bag–of–words* (Harris, 1954), caso particular que ocurre cuando los pesos de las máscaras multiplicativas toman el valor 1. Sin embargo, esta codificación entrega la flexibilidad necesaria para valorar de forma distinta la importancia de cada palabra en función de su ubicación dentro de la oración, si es que el modelo lo estima conveniente.

3.1.2. Módulo de memoria

El módulo de memoria recibe los vectores s_t, procesándolos secuencialmente y actualizando los bloques de memoria correspondientes. La memoria dinámica del modelo consiste en pares $(w_j, h_j), j \in \{1, \ldots, M\}$, donde w_j y h_j corresponden a la llave y valor correspondiente a la j-ésima memoria, respectivamente, y M es el número de bloques de memoria. Los w_j, que son parte del conjunto de pesos entrenables del modelo, permanecen constantes a lo largo del procesamiento de la secuencia, mientras que los h_j se actualizan en cada instante de tiempo, de acuerdo a un comportamiento inspirado en una red recurrente con compuerta. En particular, las ecuaciones de actualización de las memorias dinámicas se describen a continuación.

$$g_j = \sigma(s_t^{\mathrm{T}} w_j + s_t^{\mathrm{T}} h_j + b_j) \qquad g_j, b_j \in \mathbb{R}; w_j, h_j \in \mathbb{R}^e \tag{3.2}$$

$$h_j^{\sim} = \phi(U h_j + V w_j + W s_t + b_h) \qquad h_j^{\sim}, b_h \in \mathbb{R}^e; U, V, W \in \mathbb{R}^{e \times e} \tag{3.3}$$

$$h_j = h_j + g_j \odot h_j^{\sim} \qquad h_j \in \mathbb{R}^e \tag{3.4}$$

$$h_j = \frac{h_j}{\|h_j\|} \qquad h_j \in \mathbb{R}^e \tag{3.5}$$

σ y ϕ representan funciones de activación sigmoidal y PReLU (He, Zhang, Ren, & Sun, 2015), respectivamente. El valor de g_j es un escalar que actúa como mecanismo de compuerta y determina cuánto se debe actualizar la j-ésima memoria en el instante de tiempo actual. Para determinar cuánto se debe abrir la compuerta, se observa la similitud del vector de entrada s_t y los vectores w_j y h_j, por separado. A continuación, un vector candidato $h_j^{\sim}$ es computado de acuerdo a transformaciones lineales de h_j, w_j y s_t que son sumadas y activadas por ϕ. El vector candidato $h_j^{\sim}$ es usado para actualizar el valor de la memoria h_j ponderando todas sus componentes por el escalar de compuerta g_j. Finalmente, se deja al vector actualizado h_j dentro de la esfera unitaria dividiéndolo por su norma L2, lo que permite ir olvidando información no relevante.

3.1.3. Módulo de salida original

El módulo de salida original computa, mediante mecanismos de atención sobre las memorias resultantes, la respuesta a alguna representación q de una consulta. Una consulta corresponde a una secuencia de palabras $\{e_1, \ldots, e_L\}$ que es procesada a través de un mecanismo similar al módulo de entrada. Para calcular la representación q de la consulta, se computa la suma de los *embeddings* de cada e_i ponderados por máscaras multiplicativas $\{f_1, \ldots, f_L\}$. Es importante hacer notar que estas máscaras multiplicativas son diferentes a las usadas en el módulo de entrada.

Una vez que se calcula la representación de la consulta q es posible realizar la inferencia de la respuesta a ésta, usando la información de las memorias, mediante las siguientes ecuaciones.

$$p_j = \text{Softmax}(q^{\mathsf{T}} h_j) \qquad p_j \in \mathbb{R}; q \in \mathbb{R}^e \tag{3.6}$$

$$u = \sum_j p_j h_j \qquad u \in \mathbb{R}^e \tag{3.7}$$

$$y = \text{Softmax}(R\phi(q + Hu)) \qquad y \in \mathbb{R}^V; R \in \mathbb{R}^{V \times e}; H \in \mathbb{R}^{e \times e} \tag{3.8}$$

Se observa el cálculo de puntajes p_j mediante un mecanismo de atención de producto punto entre la representación de la consulta q y la memoria h_j. Luego, se computa un vector u, que corresponde a un resumen de las memorias, enfocando la atención en aquellas que son relevantes para responder la consulta. Finalmente, se obtiene una distribución de probabilidades sobre el vocabulario de respuestas con la respuesta a la consulta q.

Respecto a la formulación anterior, en el trabajo presentado en esta tesis se proponen nuevas operaciones en el módulo de salida. Las modificaciones realizadas permiten implementar métodos de atención temporal, a fin de agregar atributos de explicabilidad y mejorar el rendimiento del modelo original.

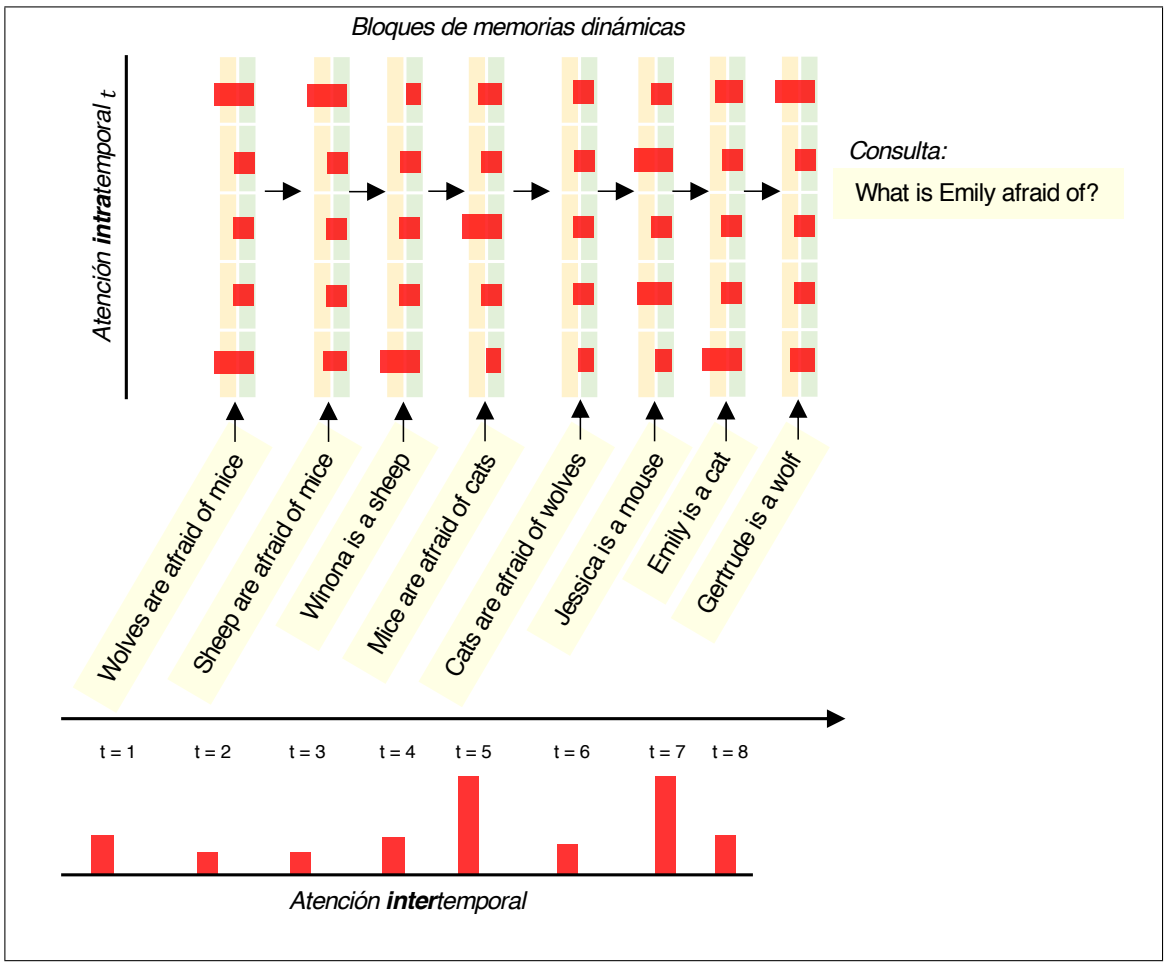

Figura 3.2. Conceptos de atención intra e intertemporal. Mediante la inspección de los valores de la distribución de atención intertemporal, se puede determinar la relevancia de cada uno de los elementos de la secuencia de entrada.

3.2. Atención intra e intertemporal

Antes de definir los módulos propuestos para generar explicaciones, se introducirán los conceptos de atención intra e intertemporal, ilustrados en la figura 3.2. Sean $\{s_1, \ldots, s_L\}$ una secuencia de vectores de características con la información necesaria para responder una consulta q, h_{tj} los estados ocultos del j-ésimo bloque de memoria luego de procesar s_t, y w_j la llave del j-ésimo bloque de memoria.

Se define la atención intratemporal como alguna función f tal que intra–attention$_{tj} = f(h_{tj}, w_j, q)$. La función f produce una distribución de atención sobre los bloques de memoria dinámica. Esta atención intratemporal permite poner énfasis en aquellos bloques de memoria relevante para responder la consulta q, para cada instante de tiempo t.

La atención intertemporal se define como alguna función g tal que inter–attention$_{tj} = g(v_t, q)$. El argumento v_t que recibe la función corresponde a un resumen de la memoria dinámica luego de realizar la suma de los bloques de memoria ponderados por la atención intratemporal en el instante t. La atención intertemporal se puede interpretar como una distribución de atención sobre los instantes de tiempo de la secuencia de entrada. Esta atención es relevante, dado que se puede supervisar, incorporándola en la función de pérdida del modelo y considerando como *ground truth* los hechos relevantes de la historia. Además, los valores que toma la atención intertemporal pueden ser interpretados como una explicación a la respuesta inferida por el modelo, debido a que va a apuntar a los instantes de tiempo t que el modelo usó para responder la consulta q.

3.3. Módulo de atención temporal pre hoc

El módulo de atención temporal pre hoc, ilustrado en la figura 3.3 realiza el cómputo de las atenciones temporales antes de realizar la inferencia de la respuesta. Es conveniente hacer notar que el modelo original solo usa las memorias h_j almacenadas luego de procesar la secuencia por completo. El módulo pre hoc hace uso de la totalidad de las memorias h_{tj}, es decir, es posible acceder a los valores de la memoria h_j luego de procesar el t-ésimo hecho de la secuencia. Este módulo tiene una gran incidencia en la tarea de *question answering* debido a que, en el grafo de cómputo, sus operaciones se encuentran inmediatamente aguas arriba de las operaciones del cálculo de respuesta. El módulo de atención temporal pre hoc reemplaza por completo el módulo de salida original de la *Entity Network*. Específicamente implementa las siguientes operaciones:

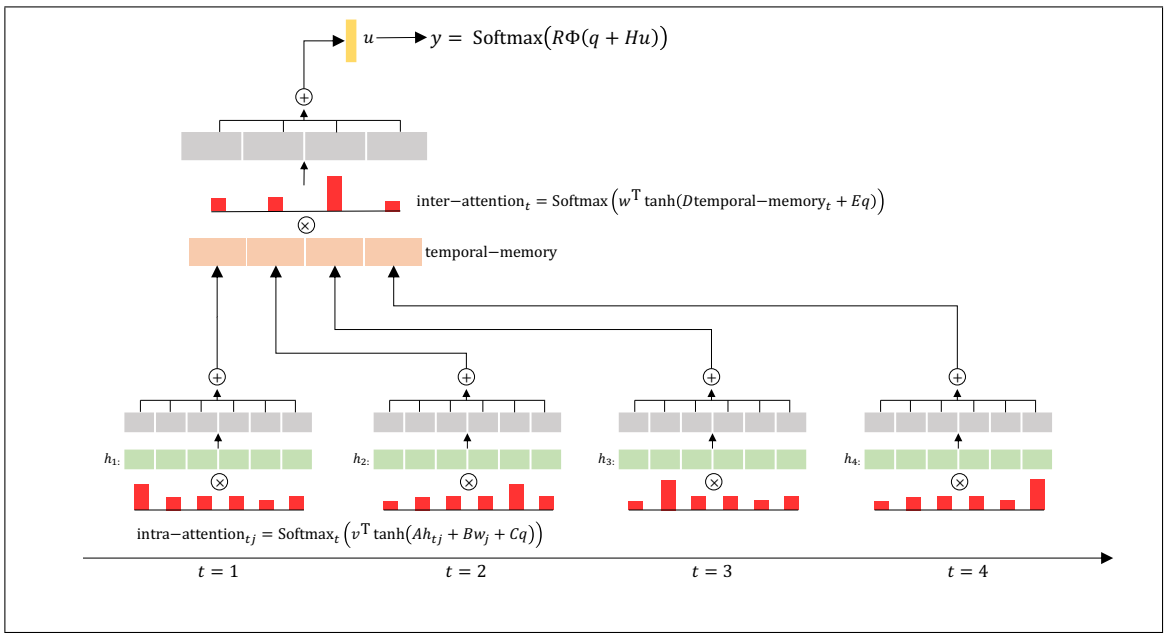

Figura 3.3. Funcionamiento del módulo de atención temporal pre hoc. Este módulo reemplaza por completo al módulo de salida original del modelo.

$$\text{intra–alignment}_{tj} = v^{\text{T}}\tanh(Ah_{tj} + Bw_j + Cq) \tag{3.9}$$

$$\text{intra–attention}_{tj} = \text{Softmax}_t(\text{intra–alignment}_{tj}) \tag{3.10}$$

$$\text{temporal–memory}_t = \sum_j \text{intra–attention}_{tj} h_{tj} \tag{3.11}$$

$$\text{inter–alignment}_t = w^{\text{T}}\tanh(D\text{temporal–memory}_t + Eq) \tag{3.12}$$

$$\text{inter–attention}_t = \text{Softmax}(\text{inter–alignment}_t) \tag{3.13}$$

$$u = \sum_t \text{inter–attention}_t \text{temporal–memory}_t \tag{3.14}$$

$$y = \text{Softmax}(R\phi(q + Hu)) \tag{3.15}$$

A continuación se indican las dimensionalidades de las operaciones y vectores anteriormente formulados.

$$\begin{aligned}
\text{intra–alignment}_{tj} &\in \mathbb{R}; v \in \mathbb{R}^m; A, B, C \in \mathbb{R}^{m \times e} \\
\text{intra–attention}_{tj} &\in \mathbb{R} \\
\text{temporal–memory}_t &\in \mathbb{R}^e \\
\text{inter–alignment}_t &\in \mathbb{R}; w \in \mathbb{R}^m; D, E \in \mathbb{R}^{m \times e} \\
\text{inter–attention}_t &\in \mathbb{R} \\
u &\in \mathbb{R}^e \\
y &\in \mathbb{R}^V; R \in \mathbb{R}^{V \times e}; H \in \mathbb{R}^{e \times e}
\end{aligned}$$

Se observa la operación de este módulo en tres etapas. En primer lugar, el cálculo de la atención intratemporal usada para formar vectores temporal–memory$_t$ que resumen las memorias, condicionadas a la consulta q, en cada instante de tiempo t. A continuación, el cálculo de la atención intertemporal usada para computar el vector u correspondiente a un resumen de la totalidad de la memoria, nuevamente condicionada a la consulta q. Finalmente, la inferencia de la respuesta mediante una operación similar al modelo original.

Respecto al cálculo de la atención intratemporal, en primer lugar se evalúa la alineación del contenido de las memorias h_{tj} y sus llaves w_j con el vector de consulta q. Esto se realiza mediante el uso de atención aditiva como se observa en la ecuación 3.9. Luego, las alineaciones son activadas mediante una función *softmax*, lo que genera una distribución de atención intratemporal sobre cada una de las memorias para cada instante de tiempo t. La distribución de atención es utilizada para calcular vectores de resumen temporal–memory$_t$, lo que es mostrado en la ecuación 3.11.

El segundo paso consiste en determinar las alineaciones intertemporales entre el contenido de temporal–memory$_t$ y el vector de consulta q. Al igual que el mecanismo de atención intratemporal, el cálculo de esta alineación se realiza mediante mecanismos de atención aditiva. Luego, se activan las alineaciones mediante una función *softmax*

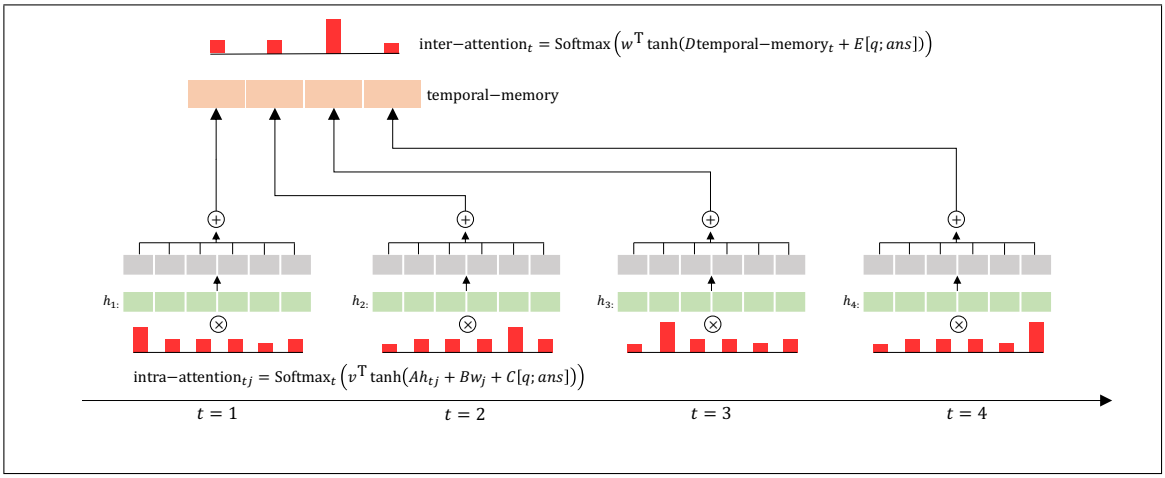

Figura 3.4. Funcionamiento del módulo de atención temporal post hoc. Este módulo opera a continuación del módulo de salida original del modelo.

generando una distribución de atención intertemporal para los distintos instantes de tiempo t, como se muestra en la ecuación 3.13.

El último paso del módulo es resumir la información de las memorias en un vector u mediante la suma de los vectores temporal–memory$_t$, ponderados por cada uno de los escalares intertemporal–attention$_t$. Este vector es llevado a un espacio de características de dimensionalidad igual al tamaño del vocabulario de posibles respuestas, como se observa en la ecuación 3.15, para formar una distribución de probabilidades de respuestas.

3.4. Módulo de atención temporal post hoc

Este módulo extiende el módulo de salida original, dejando intactas las operaciones para realizar la inferencia de la respuesta. El módulo de atención temporal post hoc, a diferencia del pre hoc, realiza el cálculo de las atenciones intertemporales después de hacer la inferencia de la respuesta. Las operaciones de este módulo se ilustran en la figura 3.4. Dicho de otra forma, implementa cálculos que en el grafo de cómputo están aguas abajo respecto a las operaciones de inferencia de respuesta.

$$\text{intra–alignment}_{tj} = v^{\mathrm{T}}\tanh(Ah_{tj} + Bw_j + C\text{concat}(q, ans)) \tag{3.16}$$

$$\text{intra–attention}_{tj} = \text{Softmax}_t(\text{intra–alignment}_{tj}) \tag{3.17}$$

$$\text{temporal–memory}_t = \sum_j \text{intra–attention}_{tj} h_{tj} \tag{3.18}$$

$$\text{inter–alignment}_t = w^{\mathrm{T}}\tanh(D\text{temporal–memory}_t + E\text{concat}(q, ans)) \tag{3.19}$$

$$\text{inter–attention}_t = \text{Softmax}(\text{inter–alignment}_t) \tag{3.20}$$

Las dimensionalidades de las matrices y vectores involucrados en las operaciones anteriormente señaladas se detallan a continuación.

$$\text{intra–alignment}_{tj} \in \mathbb{R}; v \in \mathbb{R}^m; A, B, C \in \mathbb{R}^{m \times 2e}$$

$$\text{intra–attention}_{tj} \in \mathbb{R}$$

$$\text{temporal–memory}_t \in \mathbb{R}^e$$

$$\text{inter–alignment}_t \in \mathbb{R}; w \in \mathbb{R}^m; D, E \in \mathbb{R}^{m \times 2e}$$

$$\text{inter–attention}_t \in \mathbb{R}$$

Se sigue el mismo espíritu de operación que el módulo de atención temporal pre hoc. Esencialmente, mediantes mecanismos de atención aditiva a lo largo del tiempo se infieren las relevancias de los distintos instantes temporales para responder la consulta q.

Una diferencia significativa del módulo de atención temporal post hoc, respecto a su par pre hoc, es que el primero no es responsable de inferir la distribución de probables respuestas a la consulta q. La inferencia de la respuesta es realizada de forma normal por el módulo de salida original del modelo con memoria externa.

Otra diferencia importante, relacionada con el punto anterior, es que al momento de calcular las alineaciones intra e intertemporales, el módulo de atención post hoc es capaz de integrar la información de la respuesta inferida ans. De esta forma, se apoya la tarea de inferencia de distribución de atención temporal.

Mediante el uso del módulo de atención temporal post hoc se espera agregar atributos de explicabilidad y mejorar el rendimiento en la tarea de *question answering*. Los valores de la atención intertemporal son interpretados como la probabilidad de que en el tiempo t se haya procesado un hecho relevante para responder la consulta q, lo que hace que el funcionamiento del modelo sea más interpretable. Por otra parte, al realizar la tarea de identificación de hechos relevantes se apoya el aprendizaje de mejores representaciones internas que permitan transferir conocimiento útil para la tarea de *question answering*.

4. EXPERIMENTOS, RESULTADOS Y ANÁLISIS

4.1. bAbI tasks

Ambos módulos serán evaluados en el conjunto de datos de las tareas bAbI. Las tareas bAbI son una colección de veinte tareas de *question answering*, donde cada una pone a prueba algún tipo de razonamiento cognitivo diferente. El conjunto de datos es construido de forma sintética, mediante una simulación de entidades que recorren distintos ambientes e interactúan entre si alterando el estado del mundo artificial.

Cada tarea consiste en varios ejemplos de entrenamiento. Cada ejemplo de entrenamiento consiste en:

- Una historia o secuencia de hechos $\{\text{fact}_1, \ldots, \text{fact}_T\}$. A su vez, cada hecho consiste en una secuencia de *tokens* $\{x_1, \ldots, x_{L_f}\}$.
- Una pregunta q que consiste en otra secuencia de *tokens* $\{x_1, \ldots, x_{L_q}\}$.
- Una respuesta ans.
- Los hechos relevantes supporting–facts, correspondientes al subconjunto de hechos que permiten responder la consulta sin más información.

Se puede observar un ejemplo de entrenamiento de este conjunto de datos en la figura 4.1.

Usualmente, los modelos diseñados para resolver esta clase de tareas siguen dos estrategias de entrenamiento: una débilmente supervisada, donde solo se supervisa mediante la respuesta correcta la tarea de *question answering*, y otra fuertemente supervisada, donde adicionalmente se supervisa el modelo con los hechos relevantes. El módulo de atención temporal *pre hoc* puede ser configurado para funcionar de ambas maneras. Por otra parte, el módulo de atención temporal *post hoc* opera exclusivamente en la modalidad fuertemente supervisada.

1 The suitcase is bigger than the container.
2 The container fits inside the box.
3 The chest is bigger than the chocolate.
4 The suitcase fits inside the box.
5 The chest fits inside the box.
Consulta: Does the chocolate fit in the box?
Respuesta: yes

Figura 4.1. Ejemplo de entrenamiento de *bAbI task* #18. Esta tarea requiere que el modelo desarrolle capacidades de razonamiento sobre el tamaño de entidades. Se observa la secuencia de hechos $\{\text{fact}_1, \ldots, \text{fact}_5\}$ en lenguaje natural, la consulta q respecto a esta información y la respuesta ans a dicha consulta. De color rojo, se muestra el subconjuto de hechos relevantes supporting–facts.

Las tareas bAbI se publican en dos versiones: la *1k* con 1.000 ejemplos de entrenamiento y la *10k* con 10.000 ejemplos de entrenamiento para cada tarea. Ambas versiones del conjunto de datos son acompañadas por un conjunto de datos de *test* con el mismo número de instancias que el conjunto de entrenamiento. La versión *10k* del conjunto de datos ha sido resuelta con modelos débilmente supervisados, mientras que la versión *1k* del conjunto de datos continúa siendo un desafío para estos modelos. En general, se considera que la versión *1k* del conjunto de datos es más difícil de resolver que la versión *10k* debido a que ofrece un orden de magnitud menos de ejemplos de entrenamientos para aprender los distintos tipos de razonamientos necesarios para resolver las tareas.

Los modelos presentados en esta tesis resuelven tareas de *question answering* e identificación de hechos relevantes sobre cada una de las bAbI tasks. La tarea de *question answering* se resuelve mediante los mecanismos de inferencia de respuesta, mientras que la identificación de hechos relevantes mediante la generación de distribuciones de atención intertemporal. Esta última tarea es la que agrega atributos de explicabilidad a los modelos.

4.2. Función de pérdida e hiperparámetros

Para supervisar la tarea de *question answering* se utiliza una función de pérdida $\mathcal{L}_{\text{qa}}(D, \Theta)$ de entropía cruzada, donde D es algún conjunto de datos y Θ corresponde a los pesos que parametrizan el modelo. Por otra parte, la inferencia de hechos relevantes, o *supporting facts*, se supervisa mediante una función de pérdida $\mathcal{L}_{\text{sf}}(D, \Theta)$ que mide la entropía cruzada binaria entre los hechos relevantes dados por el conjunto de datos D y la distribución de atención intertemporal inferida por el modelo parametrizado por Θ.

Al analizar el conjunto de datos, en la mayoría de las *bAbI tasks* se detecta un desbalance de clases en la tarea de identificación de hechos relevantes. En general, las secuencias de hechos de cada instancia del conjunto de datos se compone principalmente de hechos no relevantes. Dado lo anterior, y para facilitar el aprendizaje de la red, los términos de la función de pérdida $\mathcal{L}_{\text{sf}}(D, \Theta)$ asociados a los ejemplos de hechos relevantes son ponderados por un escalar $\frac{n_{\text{neg}}}{n_{\text{pos}}}$, donde n_{neg} y n_{pos} corresponden al número de hechos no relevantes y relevantes, respectivamente, en la *bAbI task* sobre la cual se está entrenando.

La función de pérdida de los modelos consiste en la suma de las funciones de pérdida de cada tarea, como se muestra en la siguiente ecuación:

$$\mathcal{L}_{\text{joint}}(D, \Theta) = \lambda_{\text{qa}}\mathcal{L}_{\text{qa}}(D, \Theta) + \lambda_{\text{sf}}\mathcal{L}_{\text{sf}}(D, \Theta) \tag{4.1}$$

Donde λ_{qa} y λ_{sf} son hiperparámetros escalares que ponderan la importancia de las tareas de *question answering* y hechos relevantes, respectivamente. A menos que se indique lo contrario, ambos hiperparámetros toman el valor 1, es decir, se consideran con igual peso en la función de pérdida.

Se observa una configuración de entrenamiento de *multitasking*, donde dos tareas se entrenan en paralelo. Se espera que la propagación de los gradientes de ambas pérdidas

respecto a los parámetros permita entregar una supervisión más rica, a fin de aprender mejores representaciones que logren incrementar el rendimiento total del modelo.

Para evaluar el rendimiento de los modelos en la tarea de *question answering* se utiliza la métrica de exactitud. Esencialmente, se comparan las respuestas correctas dadas por el conjunto de datos *versus* las respuestas inferidas, que van a estar dadas por las respuestas con mayor probabilidad en la distribución de probabilidades estimada por el modelo.

Por otra parte, para evaluar el rendimiento de los modelos en la tarea de hechos relevantes se utilizará la métrica F1. Esto es conveniente debido a que la cantidad de hechos no relevantes es muy superior a la cantidad de hechos relevantes. Dicho de otra forma, la distribución de las clases de relevancia es muy desbalanceada. La métrica se calcula comparando los hechos relevantes entregados por el conjunto de datos *versus* las atenciones intertemporales calculadas por el modelo.

Cada entrenamiento se realiza durante 200 épocas de datos. Al final de cada época se evalúa el valor de la función de pérdida en un conjunto de datos de validación, construido mediante una muestra aleatoria del 10% del conjunto de entrenamiento. Finalmente, se evalúa el conjunto de datos de *test* con el modelo que minimiza la función de pérdida sobre el conjunto de datos de validación. Como es usual en la literatura revisada, y debido a la alta variabilidad entre distintas ejecuciones de algunas tareas, cada experimento se repite 10 veces y se reporta el que entrega mejores resultados.

La tasa de aprendizaje fue variando de forma cíclica (Smith, 2017) entre 5×10^{-3} y 5×10^{-5} con un periodo de 6 épocas de entrenamiento. La dimensionalidad de los *embeddings* utilizada fue de 100. Además, se habilitaron 20 bloques de memoria dinámica. Durante los experimentos, el hiperparámetro del tamaño resultante luego de aplicar las transformaciones observadas en las ecuaciones 3.9, 3.12, 3.16 y 3.19 fue igual a 50.

Todos los parámetros son inicializados a partir de una distribución normal con media 0 y desviación estándar 1, a excepción de las pendientes de la función de activación

PReLU y las máscaras multiplicativas que se inicializan igual a 1. Para prevenir gradientes con valores muy altos, se empleó una técnica de *gradient clipping* permitiendo que éstos tuviesen a lo más norma L2 igual a 40. Finalmente, se entrenó mediante descenso de gradiente estocástico utilizando un tamaño de *batch* igual a 32 y el optimizador Adam (Kingma & Ba, 2014).

4.3. Módulo pre hoc

4.3.1. Entrenamiento asistido

Durante los experimentos de entrenamiento asistido la inferencia de la atención intertemporal, observada en la ecuación 3.13, no es realizada por el modelo. En lugar de ser computada por el modelo, la atención intertemporal tiene el valor de los hechos relevantes dados por el dataset. Lo anterior, se formaliza matemáticamente de la siguiente forma:

$$\text{inter–attention}_t = \begin{cases} 1 & \text{si fact}_t \in \text{supporting–facts} \\ 0 & \text{si fact}_t \notin \text{supporting–facts} \end{cases} \tag{4.2}$$

Debido a que los valores de la atención intertemporal son dados, el modelo no aprende a identificarlos. Este experimento es un buen ejercicio para determinar una aproximación de la cota superior de rendimiento en la tarea de responder preguntas en el caso que el módulo logre aprender de forma perfecta a identificar los hechos relevantes de una historia. De esta forma, se podrá determinar si el enfoque utilizado es apropiado o no.

Los resultados del experimento se pueden observar en la tabla 4.1. En primer lugar, se observa una diferencia significativa de rendimiento entre métodos débilmente supervisados, como la *Entity Network*, y métodos fuertemente supervisados, como la MemNN SS y el módulo pre hoc cuando $\lambda_{\text{sf}} \neq 0$. En particular, se observan incrementos sustanciales en el rendimiento en las tareas 2, 3, 14, 15 y 16. Por otra parte, existen algunas tareas,

Tabla 4.1. Resultados del entrenamiento asistido del módulo pre hoc en el conjunto de *test*. Se reportan errores de exactitud en *question answering* para cada tarea. Menos es mejor. EntNet y MemNN SS corresponden a la *Entity Network* y *End–to–End Memory Network* fuertemente supervisada, respectivamente.

Tarea	**Nombre**	**EntNet**	**MemNN SS**	**Módulo pre hoc**
1	QA con un hecho relevante	0,7%	0,0%	0,00%
2	QA con dos hechos relevantes	56,4%	0,0%	0,00%
3	QA con tres hechos relevantes	69,7%	0,0%	0,00%
4	Relaciones entre dos entidades	1,4%	0,0%	0,00%
5	Relaciones entre tres entidades	4,6%	2,0%	0,88%
6	Preguntas sí/no	30,0%	0,0%	0,00%
7	Conteo	22,3%	15,0%	9,08%
8	Listas y conjuntos	19,2%	9,0%	6,45%
9	Negación simple	31,5%	0,0%	0,00%
10	Conocimiento indefinido	15,6%	2,0%	2,83%
11	Coreferencia	8,0%	0,0%	0,00%
12	Conjunción	0,8%	0,0%	0,00%
13	Coreferencia compuesta	9,0%	0,0%	0,00%
14	Manipulación temporal	62,9%	1,0%	0,00%
15	Deducción	57,8%	0,0%	0,00%
16	Inducción	53,2%	0,0%	0,00%
17	Razonamiento posicional	46,4%	35,0%	39,45%
18	Razonamiento espacial	8,8%	5,0%	7,13%
19	Búsqueda	90,4%	64,0%	55,86%
20	Motivaciones	2,6%	0,0%	0,00%
	Media	29,6%	6,7%	6,1%
	Falladas	15	5	5

como la número 18, donde agregar supervisión a los valores de la atención intertemporal no parece ser un mecanismo útil para mejorar el rendimiento.

La tarea número 18 corresponde a una que pone a prueba la capacidad de razonamiento espacial del modelo. Esencialmente, entrega una secuencia de hechos al modelo que constatan alguna relación de tamaño entre entidades, por. ej. *the chocolate fits inside the box*, para luego hacer preguntas considerando estas relaciones, por. ej. *does the box fit in the chocolate?*. Tiene sentido que la agregación del módulo de atención temporal no ayude

significativamente al rendimiento del modelo sobre esta tarea, dado que no es una tarea cuya dimensión temporal sea relevante para la solución de ésta.

Finalmente, al comparar el módulo pre hoc con la *End–to–End Memory Network* fuertemente supervisada, se observa un rendimiento medio $0{,}6\%$ superior por parte del módulo propuesto en esta tesis. No obstante, existen aún 5 tareas que ninguno de los modelos fuertemente supervisados considerados son capaces de resolver, específicamente las tareas 7, 8, 17, 18 y 19. Como se comentó anteriormente sobre la tarea 18 de razonamiento espacial, una hipótesis razonable para explicar el desempeño insatisfactorio del modelo en estas tareas es que corresponden a tipos de razonamiento que necesitan atributos que el modelo no tiene. Por ejemplo, así como la tarea 18 correspondía a desafíos de razonamiento espacial, la tarea 17 corresponde a instancias que necesitan que el modelo sea capaz de relacionar posicionalmente a las entidades para ser resueltas correctamente. Es un desafío a futuro incorporar estructuras al modelo para adquirir esta capacidad de razonamiento.

Luego de analizar los resultados, es claro que si el módulo pre hoc es capaz de identificar correctamente los hechos relevantes de una historia, será capaz de aprender formas de razonar que los modelos débilmente supervisados no son capaces de realizar.

4.3.2. Entrenamiento parcial

Los experimentos de entrenamiento parcial se caracterizan por realizar las inferencias de hechos relevantes y de respuestas por separado. Si bien la inferencia de los hechos relevantes se realiza, la distribución de atención intertemporal no se utiliza para resumir la información de las memorias en un vector u, como se muestra en la ecuación 3.14. La inferencia de hechos relevantes, a través de la distribución de atención temporal, será solamente una salida del modelo que no será utilizada por otras operaciones del grafo de cómputo.

Es necesario modificar el módulo para poder realizar la inferencia de la respuesta correcta sin utilizar la información de la atención intertemporal. Para estos efectos, se calcula una distribución de atención sobre las memorias finales para formar un resumen u de éstas, con operaciones análogas a las ecuaciones originales de la *Entity Network* 3.6 y 3.7. Finalmente, este vector de resumen u es transformado de acuerdo a la ecuación 3.15 para inferir la respuesta a la consulta.

De acuerdo a la configuración de la función de pérdida del modelo en este experimento, es posible poner a prueba dos hipótesis:

(i) Cuando $\lambda_{\text{qa}} = 0$ y $\lambda_{\text{sf}} = 1$ en la ecuación 4.1, la función exclusiva del modelo es inferir los hechos relevantes, sin la dificultad de tener que aprender la tarea de inferir respuestas en paralelo. Bajo esta configuración es posible evaluar la hipótesis de que el modelo es capaz de inferir los hechos relevantes de una historia.

(ii) Cuando $\lambda_{\text{qa}} = \lambda_{\text{sf}} = 1$ en la ecuación 4.1, el modelo aprende en paralelo a inferir respuestas y hechos relevantes. En esta configuración se evalúa la hipótesis de que un enfoque *multitasking* mejora el rendimiento del modelo.

No es de interés evaluar el modelo cuando $\lambda_{\text{qa}} = 1$ y $\lambda_{\text{sf}} = 0$ debido a que el modelo sería prácticamente idéntico a la *Entity Network* original, cuyos resultados ya se poseen.

Los resultados mostrados en la tabla 4.3 corresponden al rendimiento del módulo pre hoc parcial sobre la tarea de identificar hechos relevantes cuando el experimento es ejecutado con $\lambda_{\text{qa}} = 0$ y $\lambda_{\text{sf}} = 1$. Es importante notar que, en promedio, al módulo pre hoc parcial le resulta más fácil identificar hechos relevantes cuando ésta es su única responsabilidad, alcanzando un puntaje F1 de $60{,}3\%$. Por el contrario, al requerir que el modelo también sea capaz de responder preguntas, es decir, cuando $\lambda_{\text{qa}} = 1$, el rendimiento en la tarea de identificar hechos relevantes cae $3{,}1$ puntos porcentuales hasta un puntaje F1 de $57{,}2\%$. Es posible concluir que la incorporación de la tarea de responder preguntas

Tabla 4.2. Resultados del entrenamiento parcial del módulo pre hoc con $\lambda_{\text{qa}} = \lambda_{\text{sf}} = 1$ en el conjunto de *test*. Se reportan errores de exactitud en *question answering* para cada tarea. Menos es mejor. EntNet corresponde a la *Entity Network*.

Tarea	**Nombre tarea**	**EntNet**	**Pre hoc parcial**
1	QA con un hecho relevante	0,7%	0,0%
2	QA con dos hechos relevantes	56,4%	64,6%
3	QA con tres hechos relevantes	69,7%	75,5%
4	Relaciones entre dos entidades	1,4%	1,2%
5	Relaciones entre tres entidades	4,6%	7,5%
6	Preguntas sí/no	30,0%	5,1%
7	Conteo	22,3%	18,1%
8	Listas y conjuntos	19,2%	3,7%
9	Negación simple	31,5%	3,2%
10	Conocimiento indefinido	15,6%	5,2%
11	Coreferencia	8,0%	6,7%
12	Conjunción	0,8%	0,7%
13	Coreferencia compuesta	9,0%	5,6%
14	Manipulación temporal	62,9%	32,8%
15	Deducción	57,8%	56,2%
16	Inducción	53,2%	52,1%
17	Razonamiento posicional	46,4%	43,3%
18	Razonamiento espacial	8,8%	8,1%
19	Búsqueda	90,4%	87,5%
20	Motivaciones	2,6%	1,7%
	Media	29,6%	23,9%
	Falladas	15	14

tiene un impacto negativo en la tarea de identificar hechos relevantes, al realizarse en paralelo. Cabe destacar que, al contrario de la tendencia, existen algunas tareas bAbI cuyo rendimiento se ve afectado de forma positiva al incorporar la tarea de responder preguntas, como por ejemplo las de razonamiento sobre preguntas sí/no (núm. 6), conteo (núm. 7) y listas y conjuntos (núm. 8).

Los resultados del experimento cuando $\lambda_{\text{qa}} = \lambda_{\text{sf}} = 1$ en la tarea de *question answering* se muestran en la tabla 4.2. Estos resultados muestran el beneficio en el rendimiento de la tarea de responder preguntas al incorporar de forma paralela un sesgo inductivo para

Tabla 4.3. Resultados del entrenamiento parcial del módulo pre hoc con $\lambda_{\text{qa}} = 1$ y $\lambda_{\text{qa}} = 0$ en el conjunto de *test*. Se reportan los puntajes F1 para cada tarea. Más es mejor. Para ambas configuraciones $\lambda_{\text{sf}} = 1$.

Tarea	**Nombre tarea**	$\lambda_{\text{qa}} = 1$	$\lambda_{\text{qa}} = 0$
1	QA con un hecho relevante	68,8%	71,2%
2	QA con dos hechos relevantes	15,7%	27,8%
3	QA con tres hechos relevantes	0,8%	9,6%
4	Relaciones entre dos entidades	100,0%	100,0%
5	Relaciones entre tres entidades	77,6%	77,8%
6	Preguntas sí/no	81,9%	80,6%
7	Conteo	55,0%	52,3%
8	Listas y conjuntos	66,6%	58,9%
9	Negación simple	77,5%	79,0%
10	Conocimiento indefinido	77,0%	75,9%
11	Coreferencia	45,8%	45,6%
12	Conjunción	51,0%	56,1%
13	Coreferencia compuesta	23,1%	30,9%
14	Manipulación temporal	57,5%	57,8%
15	Deducción	66,7%	66,7%
16	Inducción	50,0%	50,0%
17	Razonamiento posicional	66,7%	66,7%
18	Razonamiento espacial	38,5%	44,8%
19	Búsqueda	24,5%	54,2%
20	Motivaciones	100,0%	100,0%
	Media	57,2%	60,3%

reconocer hechos relevantes en una historia. Se observa una disminución en la métrica de error equivalente a un $5{,}7\%$. Por lo tanto, es beneficioso para la tarea de responder preguntas incorporar en paralelo tareas de identificación de hechos relevantes. Resulta interesante que las tareas bAbI cuyo rendimiento mejora considerablemente incluyen a las 6, 7 y 8, que también tuvieron un comportamiento positivo desde la perspectiva de la tarea de identificación de hechos relevantes. El tipo de razonamiento requerido para poder resolver estas tareas bAbI se ve fuertemente beneficiado por la realización de la tarea de *question answering* e identificación de hechos relevantes.

Por otra parte, también se observa algún grado de correlación entre tareas bAbI que tuvieron un pobre desempeño en la tarea de identificar hechos relevantes con tareas bAbI

que empeoraron su rendimiento en la tarea de *question answering* comparado a la *Entity Network* original. Por ejemplo la 2 y 3, que corresponden a historias con 2 y 3 hechos relevantes, respectivamente.

4.3.3. Entrenamiento completo

Durante el entrenamiento completo el modelo pre hoc se pone a prueba sin modificaciones respecto a lo planteado en la sección 3.3. Resulta interesante evaluar la interacción entre las tareas de *question answering* e identificación de hechos relevantes. Para lograr lo anterior se ejecutarán los siguientes experimentos:

(i) Entrenar el modelo considerando una función de pérdida con valores de $\lambda_{\text{qa}} = 1$ y $\lambda_{\text{sf}} = 0$. De esta forma se evalúa la capacidad del modelo de aprender a responder preguntas e identificar hechos relevantes, sin supervisión en esta última tarea. Esta modalidad será denominada módulo pre hoc débil.

(ii) Por otra parte, es posible entrenar configurando una función de pérdida con $\lambda_{\text{qa}} = \lambda_{\text{sf}} = 1$, donde se entrega supervisión a ambas tareas. En este caso, el modelo debe aprender las dos tareas de forma simultánea.

Los resultados del modelo en la tarea de identificar los hechos relevantes de una historia, se pueden observar en la tabla 4.5. Se observa que el módulo pre hoc débil alcanza un puntaje F1 igual a 48%. Lo anterior es relevante puesto que incluso sin contar con hechos relevantes etiquetados, el modelo es capaz de aprender a identificar parte de ellos a partir de las etiquetas de la tarea de *question answering*. Al incorporar supervisión en la tarea de identificación de hechos relevantes, el puntaje F1 se incrementa $11{,}4$ puntos porcentuales hasta $59{,}4\%$.

Al evaluar el desempeño del modelo en la tarea de *question answering*, se observan los resultados reportados en la tabla 4.4. El error medio de $25{,}6\%$ alcanzado por el módulo pre hoc débil muestra que incluso sin supervisión de hechos relevantes es posible mejorar el

Tabla 4.4. Resultados del entrenamiento completo del módulo pre hoc y post hoc respondiendo consultas en el conjunto de *test*. Se reportan errores de exactitud en *question answering* para cada tarea. Menos es mejor. EntNet corresponde a la *Entity Network*.

Tarea	Nombre tarea	EntNet	Pre hoc débil	Pre hoc	Post hoc
1	QA con un hecho relevante	0,7%	0,0%	0,3%	0,0%
2	QA con dos hechos relevantes	56,4%	69,5%	70,3%	64,4%
3	QA con tres hechos relevantes	69,7%	74,5%	67,3%	75,9%
4	Relaciones entre dos entidades	1,4%	0,2%	0,0%	2,1%
5	Relaciones entre tres entidades	4,6%	20,6%	19,8%	10,7%
6	Preguntas sí/no	30,0%	23,5%	9,5%	11,6%
7	Conteo	22,3%	20,3%	17,1%	16,7%
8	Listas y conjuntos	19,2%	7,9%	6,5%	4,5%
9	Negación simple	31,5%	15,9%	8,1%	4,1%
10	Conocimiento indefinido	15,6%	25,6%	15,8%	5,9%
11	Coreferencia	8,0%	5,3%	0,4%	6,9%
12	Conjunción	0,8%	0,0%	0,5%	0,5%
13	Coreferencia compuesta	9,0%	7,5%	0,2%	6,0%
14	Manipulación temporal	62,9%	23,0%	22,6%	32,2%
15	Deducción	57,8%	29,4%	18,1%	60,5%
16	Inducción	53,2%	50,2%	52,0%	52,7%
17	Razonamiento posicional	46,4%	41,0%	41,0%	40,9%
18	Razonamiento espacial	8,8%	9,2%	4,5%	9,5%
19	Búsqueda	90,4%	88,7%	83,1%	89,8%
20	Motivaciones	2,6%	0,0%	0,0%	1,0%
	Media	29,6%	25,6%	21,9%	24,8%
	Falladas	15	16	13	14

rendimiento en la tarea de responder preguntas. Por otra parte, al incorporar la supervisión adicional se obtiene un error medio de $21{,}9\%$.

4.4. Módulo post hoc

A diferencia del módulo pre hoc, el módulo post hoc tiene las operaciones que permiten identificar los hechos relevantes de una historia aguas abajo de las operaciones utilizadas para inferir la respuesta a una consulta. Por lo tanto, la correcta identificación de

Tabla 4.5. Resultados del entrenamiento completo del módulo pre hoc y post hoc identificando hechos relevantes en el conjunto de *test*. Se reportan los puntajes F1 para cada tarea. Más es mejor.

Tarea	**Nombre tarea**	**Pre hoc débil**	**Pre hoc**	**Post hoc**
1	QA con un hecho relevante	100,0%	91,4%	96,3%
2	QA con dos hechos relevantes	4,4%	12,2%	18,3%
3	QA con tres hechos relevantes	0,1%	3,5%	0,3%
4	Relaciones entre dos entidades	75,6%	100,0%	98,0%
5	Relaciones entre tres entidades	55,3%	59,3%	81,8%
6	Preguntas sí/no	45,0%	82,0%	82,5%
7	Conteo	36,9%	56,8%	57,6%
8	Listas y conjuntos	51,6%	58,0%	66,3%
9	Negación simple	54,6%	78,1%	84,8%
10	Conocimiento indefinido	61,8%	78,9%	81,5%
11	Coreferencia	61,4%	56,1%	53,0%
12	Conjunción	100,0%	86,8%	90,0%
13	Coreferencia compuesta	59,9%	53,4%	52,2%
14	Manipulación temporal	29,4%	43,6%	55,7%
15	Deducción	47,1%	64,0%	66,7%
16	Inducción	6,7%	25,6%	49,9%
17	Razonamiento posicional	69,7%	82,1%	66,7%
18	Razonamiento espacial	2,7%	34,8%	41,7%
19	Búsqueda	0,0%	21,2%	25,7%
20	Motivaciones	97,7%	100,0%	100,0%
	Media	48,0%	59,4%	63,4%

los hechos relevantes de una historia estará condicionado a la inferencia de la respuesta en la tarea de *question answering*.

4.4.1. Entrenamiento completo

El módulo post hoc al ser entrenado de forma completa es puesto en funcionamiento sin diferencias a lo planteado en la sección 3.4. Los resultados en las tareas de *question answering* e identificación de hechos relevantes se encuentran en las tablas 4.4 y 4.5, respectivamente.

Se observa que en la tarea de responder preguntas se alcanza un error medio igual a $24{,}8\%$, correspondiente a una disminución de $4{,}8$ puntos porcentuales respecto a la métrica alcanzada por la *Entity Network* original. La intuición detrás de este incremento en el rendimiento es que la supervisión en la tarea de hechos relevantes induce a que el modelo aprenda representaciones internas más ricas en semánticas, permitiéndole mejorar el rendimiento en la tarea de *question answering*. No obstante, el módulo post hoc no logra alcanzar el mismo rendimiento que el módulo pre hoc en la tarea de responder preguntas.

Al medir el rendimiento en la tarea de identificación de hechos relevantes, el módulo post hoc alcanza un puntaje F1 igual a $63{,}4\%$. En esta tarea el módulo post hoc es superior a cualquier versión del módulo pre hoc. Visto de otra forma, si se desea privilegiar la capacidad del modelo de identificar hechos relevantes en una historia, el módulo post hoc es el más conveniente. Por otra parte, si se quiere priorizar el rendimiento en la tarea de responder preguntas, el módulo pre hoc parece ser el más indicado.

4.5. Análisis cualitativo de tarea de identificación de hechos relevantes

No se conocen trabajos que utilicen la información de los hechos relevantes y reporten alguna métrica de esta tarea que permita comparar y hacer un análisis cuantitativo sobre el rendimiento obtenido por los modelos presentados en esta tesis. No obstante lo anterior, es posible realizar un análisis cualitativo observando ejemplos del conjunto de datos y las inferencias que realiza el modelo.

Las figuras que se analizarán son instancias del conjunto de datos pertenecientes al *split* de *test*. Dicho de otra forma, son ejemplos no vistos por el modelo durante el entrenamiento de éste. El modelo usado para procesar los ejemplos corresponde al que utiliza el módulo de atención temporal pre hoc débil, por lo que la única supervisión que recibió el modelo durante el entrenamiento fue la respuesta correcta. Los hechos con fondo de color rojo son aquellos marcados en el conjunto de datos como hechos relevantes.

Hecho	Atención intertemporal
John travelled to the hallway.	0,00
Mary journeyed to the bathroom.	0,00
Daniel went back to the bathroom.	0,00
John moved to the bedroom.	0,00
John went to the hallway.	0,00
Sandra journeyed to the kitchen.	0,00
Sandra travelled to the hallway.	0,05
John went to the garden.	0,00
Sandra went back to the bathroom.	0,26
Sandra moved to the kitchen.	0,69
Where is sandra?	*Respuesta: kitchen.*
Inferencia: kitchen.	

Figura 4.2. Ejemplo de inferencia del modelo en *bAbI task* #1 de un hecho relevante.

Hecho	Atención intertemporal
Mary got the milk there.	0,00
John moved to the bedroom.	0,00
Sandra went back to the kitchen.	0,01
Mary travelled to the hallway.	0,00
John got the football there.	0,00
John went to the hallway.	0,78
John put down the football.	0,00
Mary went to the garden.	0,04
John went to the kitchen.	0,15
Sandra travelled to the hallway.	0,03
Where is the football?	*Respuesta: hallway.*
Inferencia: bedroom.	

Figura 4.3. Ejemplo de inferencia del modelo en *bAbI task* #2 de dos hechos relevantes.

En la figura 4.2 se ve un ejemplo de la *bAbI task* #1 que se caracteriza por tener un solo hecho relevante en su historia. Se observa como el módulo de atención temporal concentra su atención en el último elemento de la historia, que coincide con el hecho relevante etiquetado en el conjunto de datos.

Hecho	**Atención intertemporal**
John and mary travelled to the hallway.	0,00
Sandra and mary journeyed to the bedroom.	0,00
Mary and daniel travelled to the bathroom.	0,00
Daniel and sandra journeyed to the office.	0,00
Daniel and mary went to the bedroom.	0,00
Daniel and sandra travelled to the hallway.	0,00
Mary and sandra journeyed to the garden.	0,00
Sandra and mary travelled to the hallway.	0,00
Daniel and john journeyed to the bathroom.	0,83
Daniel and mary went back to the office.	0,16
Where is john?	*Respuesta: bathroom.*
Inferencia: bathroom.	

Figura 4.4. Ejemplo de inferencia del modelo en *bAbI task* #12 de conjunciones.

Hecho	**Atención intertemporal**
This morning mary moved to the kitchen.	0,20
This afternoon mary moved to the cinema.	0,23
Yesterday bill went to the bedroom.	0,00
Yesterday mary journeyed to the school.	0,27
Yesterday fred went back to the cinema.	0,01
Bill journeyed to the office this morning.	0,00
Mary went to the school this evening.	0,24
This afternoon bill journeyed to the kitchen.	0,00
Julie went to the office yesterday.	0,01
This morning fred journeyed to the office.	0,01
This evening fred journeyed to the school.	0,01
This afternoon fred journeyed to the bedroom.	0,01
Where was mary before the school?	*Respuesta: cinema.*
Inferencia: cinema.	

Figura 4.5. Ejemplo de inferencia del modelo en *bAbI task* #14 de manipulación temporal.

Un aspecto interesante del comportamiento del módulo es que concentra masa de la distribución de atención en elementos de la historia que, si bien no están etiquetados como hechos relevantes, involucran a la entidad principal de la consulta. Esto se puede apreciar claramente en los ejemplos de las figuras 4.2, 4.3 y 4.5. Lo anterior es un ejemplo del

Hecho	**Atención intertemporal**
Jason is thirsty.	0,01
Antoine is bored.	0,04
Jason went to the kitchen.	0,00
Antoine journeyed to the garden.	0,00
Sumit is hungry.	0,07
Antoine grabbed the football there.	0,00
Sumit went to the kitchen.	0,00
Jason took the milk there.	0,00
Sumit took the apple there.	0,00
Yann is thirsty.	0,87
Where will yann go?	*Respuesta: kitchen.*
Inferencia: kitchen.	

Figura 4.6. Ejemplo de inferencia del modelo en *bAbI task* #20 de motivaciones.

atributo interpretable que agrega el módulo de atención temporal al modelo, debido a que es posible comprobar que la red está dando mayor peso a la información que corresponde considerar.

5. CONCLUSIONES Y TRABAJO FUTURO

Dos desafíos activos en el campo de la inteligencia artificial son desarrollar modelos capaces de adquirir capacidades de razonamiento a fin de resolver tareas complejas sin necesitar grandes volúmenes de datos, y contar con modelos potentes que no sufran deterioro en sus rendimientos al agregar atributos de explicabilidad en sus operaciones.

Un enfoque adoptado usualmente en la tarea de *question answering* consiste en desarrollar modelos con memorias externas para mantener registros de los estados de las entidades y relaciones necesarias. Modelos exitosos en esta tarea han propuesto como líneas de investigación de interés el ser capaz de interpretar de mejor forma lo que éstos aprenden y desarrollar métodos para mejorar el rendimiento en conjuntos de datos con menos ejemplos de entrenamiento (Bansal et al., 2017; Henaff et al., 2016).

En el trabajo expuesto en esta tesis se muestra de forma exitosa que la incorporación de módulos de atención temporal a un modelo recurrente con memoria externa incrementa el rendimiento en tareas de *question answering* y, de forma simultánea, agrega atributos de interpretabilidad antes inexistentes en el modelo original. En la mejor configuración, se observa un error de exactitud medio igual a 21,9%, lo que corresponde a un 26% de mejora en comparación al error obtenido por el modelo base denominado *Entity Network*.

Si bien no es posible realizar un ejercicio de comparación cuantitativo en el rendimiento obtenido en la tarea de identificación de hechos relevantes, que es la que agrega la componente de explicabilidad al modelo, el análisis cualitativo de los resultados obtenidos también es satisfactorio. El modelo es capaz de apuntar hacia los hechos relevantes de una historia incluso sin supervisión.

De todas maneras, sería interesante seguir iterando el módulo de atención pre hoc a fin de mejorar su capacidad de identificar hechos relevantes. Gracias al experimento de entrenamiento asistido queda en evidencia el incremento en rendimiento que se podría lograr si es que el modelo aprendiese a identificar de forma perfecta los hechos relevantes.

En la realidad, dado que el rendimiento del modelo en esta tarea no es perfecto, aún existe potencial de mejora.

Una línea de trabajo futuro es validar los beneficios de los módulos de atención temporal con otras clases de redes neuronales de memoria externa. Debido a la naturaleza modular de las arquitecturas planteadas, es posible integrarlos a modelos clásicos como la *Memory Network* (Sukhbaatar et al., 2015) o el *Differentiable Neural Computer* (Graves et al., 2016), entre otros.

Durante el desarrollo de este trabajo, gran parte de los avances realizados en la comunidad de procesamiento de lenguaje natural se han debido a la aplicación de modelos grandes, de cientos de millones de parámetros, en conjuntos de datos masivos (Devlin et al., 2018; Z. Yang et al., 2019). En comparación, el modelo presentado en esta tesis tiene una complejidad, en término de número de parámetros, 4 órdenes de magnitud menor. Una línea de investigación interesante en el futuro es cuantificar el impacto en el rendimiento del modelo que se produciría por el efecto de escalar el número de parámetros de éste. Si se continúa esta línea, existen conjuntos de datos de mayor complejidad que pueden ser más convenientes utilizar (Z. Yang et al., 2018).

Trabajos recientes (Camburu, Shillingford, Minervini, Lukasiewicz, & Blunsom, 2019) han desafiado modelos que generan explicaciones mediante ataques adversarios, poniendo en duda su capacidad. Otra línea de trabajo futuro interesante a desarrollar es evaluar la robustez de la atención intertemporal calculada como mecanismo de explicabilidad. Si se generan de forma consistente instancias que produzcan valores de atención intertemporal errados, quedará en duda la confiabilidad de usar los puntajes de atención intertemporal para explicar las decisiones tomadas por el modelo.

REFERENCIAS

Bahdanau, D., Cho, K., & Bengio, Y. (2014, sep). Neural Machine Translation by Jointly Learning to Align and Translate. En *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings.* Recuperado de http://arxiv.org/abs/1409.0473

Bansal, T., Neelakantan, A., & McCallum, A. (2017, jun). RelNet: End-to-End Modeling of Entities & Relations. En *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA.* Recuperado de http://arxiv.org/abs/1706.07179

Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., ... Pascanu, R. (2018). *Relational inductive biases, deep learning, and graph networks.* Recuperado de http://arxiv.org/abs/1806.01261

Brynjolfsson, E., & Mitchell, T. (2017, dec). What can machine learning do? Workforce implications. *Science*, *358*(6370), 1530–1534. Recuperado de http://www.sciencemag.org/lookup/doi/10.1126/science.aap8062 doi: 10.1126/science.aap8062

Camburu, O.-M., Shillingford, B., Minervini, P., Lukasiewicz, T., & Blunsom, P. (2019). Make Up Your Mind! Adversarial Generation of Inconsistent Natural Language Explanations. En *NeurIPS 2019 Workshop on Safety and Robustness in Decision Making* (pp. 1–9). Recuperado de http://arxiv.org/abs/1910.03065

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018, oct). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.* Recuperado de http://arxiv.org/abs/1810.04805

Freitas, A. A. (2014). Comprehensible Classification Models - a position paper. *ACM*

SIGKDD Explorations Newsletter, *15*(1), 1–10. doi: 10.1145/2594473.2594475

Graves, A., Wayne, G., & Danihelka, I. (2014). *Neural Turing Machines.* Recuperado de http://arxiv.org/abs/1410.5401

Graves, A., Wayne, G., Reynolds, M., Harley, T., Danihelka, I., Grabska-Barwińska, A., … Hassabis, D. (2016). Hybrid computing using a neural network with dynamic external memory. *Nature*, *538*(7626), 471–476. doi: 10.1038/nature20101

Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., & Pedreschi, D. (2018, aug). A Survey of Methods for Explaining Black Box Models. *ACM Computing Surveys*, *51*(5), 1–42. Recuperado de http://arxiv.org/abs/1802.01933 doi: 10.1145/3236009

Harris, Z. S. (1954, aug). Distributional Structure. *WORD*, *10*(2-3), 146–162. Recuperado de http://www.tandfonline.com/doi/full/10.1080/00437956.1954.11659520 doi: 10.1080/00437956.1954.11659520

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. En *Proceedings of the IEEE International Conference on Computer Vision* (pp. 1026–1034). doi: 10.1109/ICCV.2015.123

Henaff, M., Weston, J., Szlam, A., Bordes, A., & LeCun, Y. (2016). Tracking the World State with Recurrent Entity Networks. En *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings.* Recuperado de http://arxiv.org/abs/1612.03969

Hirschberg, J., & Manning, C. D. (2015, jul). Advances in natural language processing. *Science*, *349*(6245), 261–266. Recuperado de http://www.sciencemag.org/cgi/doi/10.1126/science.aaa8685 doi: 10.1126/science.aaa8685

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780. doi: 10.1162/neco.1997.9.8.1735

Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. En *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Recuperado de http://arxiv.org/abs/1412.6980

LeCun, Y., Bengio, Y., & Hinton, G. (2015, may). Deep learning. *Nature*, *521*(7553), 436–444. Recuperado de http://www.nature.com/articles/nature14539 doi: 10.1038/nature14539

Marcus, G. (2018, jan). *Deep Learning: A Critical Appraisal* (Tech. Rep.). Recuperado de http://arxiv.org/abs/1801.00631

Mitchell, T. M. (1980). *The Need for Biases in Learning Generalizations* (Tech. Rep. No. CBM-TR-117). Recuperado de http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.19.5466

Rajpurkar, P., Jia, R., & Liang, P. (2018). Know What You Don't Know: Unanswerable Questions for SQuAD. En *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers* (pp. 784–789). doi: 10.18653/v1/p18-2124

Siegelmann, H., & Sontag, E. (1995, feb). On the Computational Power of Neural Nets. *Journal of Computer and System Sciences*, *50*(1), 132–150. Recuperado de https://doi.org/10.1006/jcss.1995.1013 doi: 10.1006/jcss.1995.1013

Smith, L. N. (2017). Cyclical learning rates for training neural networks. En *Proceedings - 2017 IEEE Winter Conference on Applications of Computer Vision, WACV 2017* (pp. 464–472). doi: 10.1109/WACV.2017.58

Sukhbaatar, S., Szlam, A., Weston, J., & Fergus, R. (2015, mar). End-To-End Memory Networks. En *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal,*

Quebec, Canada. Recuperado de http://arxiv.org/abs/1503.08895

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). *Sequence to Sequence Learning with Neural Networks.* Recuperado de http://arxiv.org/abs/1409.3215

Turing, A. M. (1950). Computing Machinery And Intelligence. *Mind*, *LIX*(236), 433–460. Recuperado de https://academic.oup.com/mind/article-lookup/doi/10.1093/mind/LIX.236.433 doi: 10.1093/mind/LIX.236.433

Vinyals, O., Fortunato, M., & Jaitly, N. (2015). *Pointer Networks.* Recuperado de http://arxiv.org/abs/1506.03134

Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2014, nov). Show and Tell: A Neural Image Caption Generator. En *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 3156–3164). Recuperado de http://arxiv.org/abs/1411.4555 doi: 10.1109/CVPR.2015.7298935

Von Neumann, J. (1993). First draft of a report on the EDVAC. *IEEE Annals of the History of Computing*, *15*(4), 27–75. Recuperado de http://ieeexplore.ieee.org/document/238389/ doi: 10.1109/85.238389

Weston, J., Bordes, A., Chopra, S., Rush, A. M., van Merriënboer, B., Joulin, A., & Mikolov, T. (2016, feb). Towards AI-Complete Question Answering: A Set of Prerequisite Toy Tasks. En *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings.* Recuperado de http://arxiv.org/abs/1502.05698

Weston, J., Chopra, S., & Bordes, A. (2015). Memory Networks. En *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings.* Recuperado de http://arxiv.org/abs/1410.3916

Yang, Y., Yih, W.-t., & Meek, C. (2015). WikiQA: A Challenge Dataset for Open-Domain

Question Answering. En *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* (pp. 2013–2018).

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., & Le, Q. V. (2019, jun). *XLNet: Generalized Autoregressive Pretraining for Language Understanding.* Recuperado de http://arxiv.org/abs/1906.08237

Yang, Z., Qi, P., Zhang, S., Bengio, Y., Cohen, W., Salakhutdinov, R., & Manning, C. D. (2018, sep). HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. En *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing* (pp. 2369–2380). Stroudsburg, PA, USA: Association for Computational Linguistics. Recuperado de http://arxiv.org/abs/1809.09600http://aclweb.org/anthology/D18-1259 doi: 10.18653/v1/D18-1259