(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 993 023**

(51) Int. CI.:

G06N 3/098 (2013.01)
G06N 3/045 (2013.01)
G06F 40/216 (2010.01)
G06F 40/30 (2010.01)
G06N 3/09 (2013.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **18.02.2016 PCT/US2016/018536**

(87) Fecha y número de publicación internacional: **25.08.2016 WO16134183**

(96) Fecha de presentación y número de la solicitud europea: **18.02.2016 E 16753087 (2)**

(97) Fecha y número de publicación de la concesión europea: **25.09.2024 EP 3259688**

(54) Título: **Sistemas y métodos para modelado de lenguaje neuronal**

(30) Prioridad:

19.02.2015 US 201562118200 P
05.03.2015 US 201562128915 P

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
20.12.2024

(73) Titular/es:

DIGITAL REASONING SYSTEMS, INC. (100.0%)
701 Cool Springs Blvd.Fifth Floor
Franklin, TN 37067, US

(72) Inventor/es:

TRASK, ANDREW;
GILMORE, DAVID y
RUSSELL, MATTHEW

(74) Agente/Representante:

ISERN JARA, Jorge

DESCRIPCIÓN

Sistemas y métodos para modelado de lenguaje neuronal

Antecedentes

Los sistemas de procesamiento de lenguaje natural (NLP) buscan automatizar la extracción de información útil a partir de secuencias de símbolos en el lenguaje humano. Algunos sistemas de NLP pueden encontrar dificultades debido a la complejidad y escasez de información en lenguaje natural. Los modelos de lenguaje de red neuronal (NNLM) pueden superar las limitaciones del rendimiento de los sistemas tradicionales. Un NNLM puede aprender representaciones distribuidas de palabras y puede incorporar un vocabulario en un espacio lineal dimensional más pequeño que modela una función de probabilidad para secuencias de palabras, expresada en términos de estas representaciones.

Los NNLM pueden generar incrustaciones de palabras al entrenar una tarea de predicción de símbolos en una ventana de contexto local en movimiento. El conjunto ordenado de pesos asociados con cada palabra se convierte en la incrustación vectorial densa de esa palabra. El resultado es un modelo de espacio vectorial que codifica relaciones semánticas y sintácticas. Un NNLM puede predecir una palabra dado su contexto circundante. Estas representaciones distribuidas codifican matices de significado a través de sus dimensiones, lo que permite que dos palabras tengan múltiples relaciones de valor real codificadas en una sola representación. Esta característica se deriva de la hipótesis distributiva: las palabras que aparecen en contextos similares tienen un significado similar. Las palabras que aparecen en contextos similares experimentarán ejemplos de entrenamiento similares, resultados de entrenamiento, y convergerán con pesos similares.

Una vez calculadas, las incrustaciones de palabras con base en analogías de palabras pueden permitir operaciones vectoriales entre palabras que reflejan sus relaciones semánticas y sintácticas. La analogía "el rey es a la reina como el hombre es a la mujer" se puede codificar en el espacio vectorial mediante la ecuación rey - reina = hombre - mujer.

Los NNLM convencionales pueden no tener en cuenta la morfología y la forma de las palabras. Sin embargo, la información sobre la estructura de las palabras en las representaciones de palabras puede ser valiosa para parte del análisis del habla, la similitud de palabras, y la extracción de información. Es con respecto a estas y otras consideraciones que se presentan aspectos de la presente divulgación en la presente. Quoc Le et al: "Distributed Representations of Sentences and Documents", 16 May 2014 (2014-05-16), propone Paragraph Vector, un algoritmo no supervisado que aprende representaciones de características de longitud fija a partir de fragmentos de texto de longitud variable, tal como oraciones, párrafos, y documentos.

Sumario

La invención se define en las reivindicaciones independientes anexas. Las reivindicaciones dependientes adjuntas definen realizaciones preferidas.

Breve descripción de las figuras

Ahora se hará referencia a las figuras anexas, que no necesariamente se dibujan a escala.

La figura 1 representa un nodo neuronal de acuerdo con una realización.

La figura 2 representa una red neuronal de acuerdo con una realización.

La figura 3 representa un modelo de lenguaje de red neuronal sin partición, utilizando una bolsa continua de arquitectura de palabras.

La figura 4 representa un modelo de lenguaje de red neuronal particionada con ventanas de acuerdo con una realización.

La figura 5 representa un modelo de lenguaje de red neuronal particionada direccional de acuerdo con una realización.

La figura 6 representa la exactitud relativa de cada partición en un modelo de PENN a juzgar por las puntuaciones de analogía de palabras relativas a filas.

La figura 7 es un diagrama de arquitectura informática de un sistema informático capaz de implementar aspectos de la presente divulgación de acuerdo con una o más realizaciones.

Descripción detallada

Aunque se explican en detalle realizaciones de ejemplo de la presente divulgación, se va a entender que se contemplan otras realizaciones. Por consiguiente, no se propone que el alcance de la divulgación se limite a los detalles de construcción y arreglo de componentes expuestos en la siguiente descripción o ilustrados en las figuras. La divulgación es capaz de otras realizaciones y de practicarse o llevarse a cabo de diversas maneras.

También se debe observar que, como se utiliza en la especificación y las reclamaciones anexas, las formas singulares "uno", "una", "el" y "la" incluyen referencias plurales salvo que el contexto dicte claramente lo contrario. Se propone que las referencias a una composición que contiene "un" constituyente incluyan otros constituyentes además del mencionado.

Además, al describir las realizaciones de ejemplo, se recurrirá a la terminología por el bien de la claridad. Se propone que cada término contemple su significado más amplio como se entiende por aquellos expertos en la técnica e incluye todos los equivalentes técnicos que operan de una manera similar para lograr un propósito similar.

Los intervalos se pueden expresar en la presente como de "alrededor de" o "aproximadamente" o "sustancialmente" un valor en particular y/o a "alrededor de" o "aproximadamente" o "sustancialmente" otro valor particular. Cuando se expresa este intervalo, otras realizaciones de ejemplo incluyen desde el valor particular y/o hasta el otro valor particular.

En la presente, el uso de términos como "que tiene", "tiene", "que incluye" o "incluye" son abiertos y pretenden tener el mismo significado que los términos como "que comprende" o "comprende" y no excluyen la presencia de otra estructura, material, o actos. De manera similar, aunque el uso de términos como "puede" o "puede" pretende ser abierto y reflejar que la estructura, el material o los actos no son necesarios, la falta de uso de dichos términos no se propone que refleje que la estructura, el material o los actos son esenciales. En la medida en que la estructura, el material o los actos se consideren actualmente esenciales, se identifican como tales.

Se va a entender también que la mención de uno o más pasos de método no excluye la presencia de pasos de método adicionales o pasos de método intermedios entre aquellos pasos expresamente identificados. Además, aunque el término "etapa" se puede usar en la presente para connotar diferentes aspectos de los métodos empleados, el término no se debe interpretar como que implica ningún orden particular entre los diversos pasos descritos en la presente, salvo y excepto cuando el orden de los pasos individuales se requiera explícitamente.

Se propone que los componentes descritos en lo sucesivo que constituyen diversos elementos de la presente divulgación sean ilustrativos y no restrictivos. Se propone que muchos componentes adecuados que realizarían las mismas o similares funciones que los componentes descritos en la presente se abarquen dentro del alcance de la presente divulgación. Estos otros componentes no descritos en la presente pueden incluir, pero no se limitan a, por ejemplo, componentes similares que se desarrollan después del desarrollo de la presente materia divulgada en la presente.

Para facilitar la comprensión de los principios y características de la presente divulgación, a continuación, se explican varias realizaciones ilustrativas. En particular, la presente materia actualmente descrita se describe en el contexto de NNLM. La presente divulgación, sin embargo, no está tan limitada, y puede ser aplicable en otros contextos. Por ejemplo, y sin limitación, algunas realizaciones de la presente divulgación pueden mejorar otras técnicas de reconocimiento de secuencias y similares. Estas realizaciones se contemplan dentro del alcance de la presente divulgación. Por consiguiente, cuando la presente divulgación se describe en el contexto de NNLM, se entenderá que otras realizaciones pueden tomar el lugar de aquellas a las que se hace referencia.

Una red neuronal puede comprender una pluralidad de capas de nodos neuronales (es decir, "neuronas"). En algunas realizaciones, una red neuronal comprende una capa de entrada, una capa oculta, y una capa de salida. Para facilitar la explicación, a continuación, se explicará el funcionamiento básico de un nodo neuronal de acuerdo con una realización. Sin embargo, como se entendería por un experto en la técnica, se podrían utilizar otros tipos de neuronas.

Las redes neuronales se implementan por computadora. La pluralidad de capas y nodos puede residir en módulos de programa ejecutables (por ejemplo, módulos de programa 714 en la figura 7) u otras construcciones de software, o en componentes de hardware programados dedicados. Las capas y nodos, y otros componentes funcionales descritos en la presente de acuerdo con diversas realizaciones para realizar aspectos de modelado de lenguaje neuronal se pueden almacenar en dispositivos de memoria (por ejemplo, memoria 704 o almacenamiento masivo 712) y ejecutables por procesadores (por ejemplo, unidad de procesamiento 702) de una o más computadoras, tal como la computadora 700 mostrada en la figura 7. El análisis, procesamiento de datos y otras funciones asociadas con la operación de las capas y nodos y la realización de las diversas funciones de modelado de lenguaje neuronal descritas en la presente, se pueden provocar por la ejecución de instrucciones por uno o más de estos procesadores. Las funciones de entrenamiento, tal como los procesos de entrenamiento modelo como se describen en la presente, se pueden realizar junto con las interacciones de uno o más usuarios con una o más computadoras, tal como la computadora 700 de la figura 7, y se puede operar y configurar tal que los modelos

entrenables se puedan mejorar con base en la interacción de los usuarios con los datos de entrenamiento y los modelos anteriores, y se pueden implementar en varios datos de acuerdo con el aprendizaje automático que se puede supervisar y/o autónomo.

La figura 1 ilustra un nodo neuronal 100 de acuerdo con una realización. En algunas realizaciones, cada nodo 100 puede tener una pluralidad de entradas 110, cada una con un peso 120, y una salida individual 130. La salida de un nodo se calcula como:

$$y_k = \varphi(\sum_{j=0} w_{kj} x_j)$$

donde la salida y del nodo k^{enésimo} se calcula como la suma de cada entrada x del nodo multiplicado por un peso correspondiente w_{kj} para esa entrada. El resultado de la suma se transforma por una función de transferencia. En algunas realizaciones, la función de transferencia *(p* puede ser una función sigmoidea, representada por:

$$S(t) = \frac{1}{1+e^{-t}}$$

En algunas realizaciones, cada entrada puede aceptar un valor entre cero y uno, aunque se pueden utilizar otros valores. Colectivamente, las entradas se pueden representar por el vector de entrada x:

$$x = [x_0\, x_1\, \ldots\, x_n]$$

Cada entrada también tiene un peso asociado con ella, con el conjunto colectivo de pesos de entrada que comprende el vector:

$$w_k = [w_0, w_1, w_2 \ldots w_m]$$

La figura 2 representa una red neuronal generalizada. Esta red comprende una capa de entrada 210 con tres nodos, una capa oculta 220 con cuatro nodos y una capa de salida 230 con tres nodos. Los nodos en la capa de entrada 210 emiten un valor, pero no calculan un valor por sí mismos. En una realización, la salida puede ser un número entre cero y uno. Cada nodo en la capa oculta 220 recibe salidas de cada nodo de la capa de entrada 210, y emite el resultado de la ecuación $y_k = \varphi(\sum_{j=0} w_{kj} x_j)$ expuesta anteriormente. La salida de la capa oculta 220 se alimenta entonces hacia adelante a la capa de salida 230. Cada nodo en la capa de salida calcula su salida como el resultado de la ecuación $y_k = \varphi(\sum_{j=0} w_{kj} x_j)$, que corresponde a la salida de la red. Esta representación de una red neuronal se propone que ayude con la comprensión de las redes neuronales y no limita la tecnología divulgada. Es decir, una red neuronal puede constar de cientos, miles, millones o más nodos en cada una de las capas de entrada, oculta, y de salida. Además, las redes neuronales pueden tener una capa oculta individual (como se representa), o pueden tener múltiples capas ocultas.

Los pesos para una capa de nodos se pueden representar además por la matriz:

$$w = [w_{0,0}\, w_{0,1} \cdots w_{0,j}\, w_{1,0}\, w_{1,1} \quad \vdots \quad \ddots \quad \vdots \quad w_{k,0} \quad \cdots \quad w_{k,j}]$$

por lo tanto, la salida para el grupo de nodos se puede calcular como:

$$y = \varphi(w \times x)$$

Se puede utilizar una red neuronal de acuerdo con algunas realizaciones para analizar una lista ordenada de unidades lingüísticas. Una unidad lingüística como se define en la presente se puede referir a una frase, palabra, letra, u otro carácter o caracteres utilizados en el idioma. La red neuronal se configura para tomar la lista ordenada de unidades lingüísticas, con una unidad lingüística omitida, y predecir la unidad lingüística omitida. Esta unidad lingüística omitida se conoce como un "término de enfoque". Por ejemplo, la figura 3 representa una red neuronal de una realización, que no está cubierta por la presente materia de las reivindicaciones y en la que la red neuronal está analizando la frase "SEE SPOT RUN". Los nodos de entrada para "SEE" y "RUN" están activados, para predecir el término de enfoque "SPOT". Entonces, la red neuronal predice adecuadamente que la palabra que falta es "SPOT" al retornar un 100% en el nodo de salida "SPOT".

La red neuronal 200 tiene una capa de entrada 210 de nodos que codifica esta entrada usando codificación "one-hot". Es decir, existe un nodo de entrada para cada unidad lingüística en un diccionario de unidades lingüísticas que se podrían utilizar. El diccionario no necesita ser un diccionario completo, tal como un diccionario de inglés completo, pero puede existir como un subconjunto de caracteres, palabras, o frases utilizadas en el idioma. Por ejemplo, si la lista ordenada de unidades lingüísticas es una palabra en el idioma inglés, el diccionario puede ser la lista de letras A-Z, todas las letras mayúsculas (A-Z) y minúsculas (a-z) y signos de puntuación, o algún subconjunto de las mismas. Si la lista ordenada de unidades lingüísticas es una frase en inglés, el diccionario puede incluir todas o algunas palabras en el idioma inglés. Por ejemplo, si el sistema se va a entrenar en un libro o corpus de texto específico, se pueden omitir las palabras en inglés que no se usan en el libro o corpus. En

algunas realizaciones, los diccionarios pueden incluir además términos compuestos que incluyen más de una palabra, tal como “San Francisco”,“hot dog”, etc.

La red neuronal tiene una capa de nodos ocultos 220. Esta capa de nodos ocultos 220 se divide en particiones. Cuando una capa de nodos ocultos se divide en particiones, se conoce como una red neuronal de incrustación particionada (PENN). Cada partición se relaciona con una posición en una frase (una palabra antes del término de enfoque, una palabra después del término de enfoque, etc.). Esto se puede conocer como incrustación con ventanas. La figura 4 representa una realización particionada con ventanas que tiene dos particiones, una para la palabra inmediatamente anterior al término de enfoque (p=+1) y otra para la palabra inmediatamente posterior al término de enfoque (p=-1). La red se muestra aquí, analizando la frase "SEE SPOT RUN", donde el término de enfoque es "SPOT". Aquí, se utilizan tres nodos ocultos para la partición p=+1, y tres nodos ocultos para la partición p=-1. De nuevo, como resultado de ingresar "SEE" a la partición p=+1, y "RUN" a la partición p=-1, la red predice que el término de enfoque es "SPOT".

En algunas realizaciones, cada partición puede referirse a una dirección en la frase (todas las palabras antes del término de enfoque, todas las palabras después del término de enfoque, etc.). Esto se puede conocer como incrustación direccional. Como se reconocería por un experto en la técnica, los enfoques se pueden combinar en numerosas permutaciones, tal como una realización que tiene una partición para la unidad lingüística dos ventanas antes del término de enfoque, y una partición para todas las unidades lingüísticas después del término de enfoque, y otras permutaciones. La figura 5 representa una realización con partición direccional que tiene una partición para todas las palabras antes del término de enfoque (P > 0), y una partición para todas las palabras después del término de enfoque (P < 0). Aquí, la red neuronal está analizando la frase "SEE SPOT RUN FAST" donde el término de enfoque es "SPOT". Como se muestra, la palabra "SEE" se ingresa en la partición P > 0, y las palabras "FAST" y "RUN" se ingresan en la partición P < 0. Entonces, la red neuronal retorna el término de enfoque "SPOT".

Cuando se utilizan particiones, se utiliza un conjunto separado de nodos de entrada para cada partición. De manera alternativa, en una realización que no se cubre por la presente materia de las reivindicaciones, el conjunto de nodos de entrada se puede modelar como nodos especializados capaces de emitir una salida separada a cada partición. Ambos son matemáticamente equivalentes. Para simplificar la comprensión, la presente descripción se refiere a conjuntos de nodos de entrada para cada partición separada, pero en cada caso en el que se utilizan nodos de entrada separados para cada partición, se podría utilizar un conjunto individual de nodos de entrada que alimentan valores diferentes a diferentes particiones en realizaciones que no están cubiertas por la presente materia de las reivindicaciones.

Como se analiza anteriormente, cada nodo en cada partición de la capa oculta tendrá un conjunto de pesos asociados con el mismo, representados por un vector x. En algunas realizaciones, el vector x tendrá tantos elementos como nodos de entrada asociados con la partición. En algunas realizaciones, algunos nodos de capa oculta en una partición determinada pueden alimentarse desde menos de todos los nodos de entrada asociados con esa partición. Esto se puede modelar ya sea como un vector x que tiene menos elementos que el número de nodos de entrada asociados con esa partición, o un vector de longitud equivalente con los pesos de términos omitidos establecidos en cero. Cada nodo oculto puede tener además un término de sesgo, un peso asociado con un nodo de entrada, que ya sea no se multiplica por nada o se multiplica por 1 y se adiciona al resultado. Los pesos para todos los nodos ocultos en una partición de la capa oculta se pueden representar por una matriz formada al concatenar los vectores para cada nodo en la capa oculta de nodos. Esta matriz se puede conocer como una matriz de "sinapsis". Debido a que es el primer conjunto de nodos de la capa de entrada, se conoce como sinapsis 0 (synO). Además, esta matriz se puede conocer como una "matriz de incrustación".

$$syn_0 = [w_1 \dots w_k] = [w_{0,0}\, w_{0,1} \cdots w_{0,j}\, w_{1,0}\, w_{1,1} \quad \vdots \quad \ddots \quad \vdots \quad w_{k,0} \cdots w_{k,j}]$$

En algunas realizaciones, sólo hay una capa oculta individual, aunque en algunas realizaciones puede haber dos o más capas ocultas. En realizaciones que tienen múltiples capas ocultas, puede haber el mismo número de particiones que la capa anterior, más o menos. En algunas realizaciones, una capa oculta adicional puede no tener particiones en absoluto.

La última capa oculta es seguida por una capa de salida. En algunas realizaciones, la capa de salida puede tener una neurona de salida asociada con cada unidad lingüística en un diccionario de unidades lingüísticas. En algunas realizaciones, hay menos conjuntos de nodos de salida que particiones en la capa oculta anterior, que toma como entradas las salidas de más de una partición en la última capa oculta de nodos. En algunas realizaciones, hay un conjunto individual de nodos de salida, cada nodo recibe como entrada la salida de todos los nodos en la capa anterior. En algunas realizaciones, hay un conjunto separado de nodos de salida asociados con cada partición en la última capa oculta de nodos.

CLOW

En algunas realizaciones, una red neuronal de la presente divulgación se puede entrenar utilizando un estilo de

entrenamiento de lista continua de palabras (CLOW). El estilo de entrenamiento de CLOW bajo el marco de PENN optimiza la siguiente función de objetivo:

$$arg \quad max_{\theta}(\prod_{(w,C)\in d}\prod_{-c\leq j\leq c, j\neq 0} p(c_j^i;\theta) \qquad \prod_{(w,C)\in d'}\prod_{-c\leq j\leq c, j\neq 0} p(w=0|c_j^i;\theta))$$

donde c_j^i es la representación específica de ubicación (partición *j)* para la palabra en la posición de ventana *j* con relación a la palabra de enfoque *w.*

En este estilo de entrenamiento, la capa de salida se configura como un conjunto individual (o una partición individual) de nodos que reciben entradas de cada nodo en la capa oculta anterior. La red neuronal se entrena entonces en un corpus lingüístico, que es una o más secuencias de una pluralidad de unidades lingüísticas. Se ejecutan uno o más ejemplos de entrenamiento, donde para cada ejemplo de entrenamiento, las palabras que circundan el término de enfoque se ingresan en sus respectivos nodos de entrada que corresponden a las particiones apropiadas. Por ejemplo, si el corpus lingüístico incluye la frase "SEE SPOT RUN FAST", y "SPOT" es el término de enfoque, los nodos de entrada, asociados con la posición uno en adelante del término de enfoque, se activarían de acuerdo con el término "SEE", y la posición uno detrás del término de enfoque se activaría de acuerdo con el término "RUN". Esas entradas se propagan a través de la red neuronal para producir una salida. La salida de la red neuronal, que se correlaciona con el porcentaje de probabilidad de que la palabra de salida sea el término de enfoque, se compara entonces con una salida preferida donde el término de enfoque ("SPOT") es 100% (o un valor de salida máximo correspondiente) y la salida para todas las demás unidades lingüísticas es 0% (o un valor de salida mínimo correspondiente). La salida real se compara a la salida preferida, y entonces se propaga hacia atrás a través de la red neuronal para actualizar las matrices de sinapsis de acuerdo con uno o más algoritmos de propagación hacia atrás de errores conocidos. Este proceso se puede repetir en secuencias adicionales en el corpus lingüístico, hasta que la red sea lo suficientemente exacta.

En algunas realizaciones, la implementación de CLOW direccional con tamaños de ventana muy pequeños (en la imagen con un tamaño de ventana de 1) se puede utilizar para lograr un rendimiento aceptable. CLOW direccional es capaz de lograr una puntuación de paridad utilizando un tamaño de ventana de 1, en contraste con word2vec que utiliza un tamaño de ventana de 10 cuando todos los demás parámetros son iguales. En algunas realizaciones, esta optimización puede reducir la cantidad de ejemplos de entrenamiento y el tiempo de entrenamiento general en un factor de 10.

SKIP-GRAM

En algunas realizaciones, una red neuronal de la presente divulgación se puede entrenar utilizando un estilo de entrenamiento de "skip-gram". El estilo de entrenamiento de skip-gram optimiza la siguiente función objetivo:

$$arg \quad max_{\theta}(\prod_{(w,C)\in d}\prod_{c\leq j\leq c, j\neq 0} p(c_j^i;\theta) \qquad \prod_{(w,C)\in d'}\prod_{c\leq j\leq c, j\neq 0} p(w_j=0|c_j^i;\theta))$$

donde c_j^i . es la representación específica de ubicación (partición *j)* para la palabra en la posición de ventana *j* con relación a la palabra de enfoque *w.*

En este estilo de entrenamiento, se configura una red que tiene particiones de salida separadas para cada partición de capa oculta. Por lo tanto, en el ejemplo listado anteriormente, la red neuronal constaría de dos redes neuronales, una que modela la probabilidad de que el término de enfoque sea un valor determinado con base en la palabra en la posición uno por delante del término de enfoque, y otra que modela la probabilidad de que el término de enfoque sea un valor determinado con base en la palabra en la posición uno después del término de enfoque. En otras palabras, si el corpus lingüístico incluye la frase "SEE SPOT RUN FAST", y "SPOT" es el término de enfoque, una partición recibirá como entrada "SEE", y una partición recibirá como entrada "RUN". Los nodos de salida correspondientes entonces generarán una salida. Esta salida real se compara con una salida preferida donde el término de enfoque ("SPOT") es 100% (o un valor de salida máximo correspondiente) y la salida para todas las demás unidades lingüísticas es 0% (o un valor de salida mínimo correspondiente). La salida real se compara a la salida preferida, y entonces se propaga hacia atrás a través de la red neuronal para actualizar las matrices de sinapsis de acuerdo con cualquier algoritmo de propagación hacia atrás de errores, como se conoce en la técnica de las redes neuronales. Este proceso se puede repetir en secuencias adicionales en el corpus lingüístico, hasta que la red sea lo suficientemente exacta. Después del entrenamiento, para llegar a un resultado final, los resultados de las diversas particiones de salida se pueden sumar o combinar de otra manera para producir una probabilidad final del término de enfoque.

En algunas realizaciones, el estilo de entrenamiento de omisión de diagrama se puede utilizar para entrenar una red neuronal en paralelo. Es decir, cada partición separada de nodos ocultos, y partición de nodos de salida se puede entrenar independientemente de otras particiones de nodos ocultos y nodos de salida. Por lo tanto, esas realizaciones podrían entrenarse usando múltiples subprocesos en un procesador de computadora individual, o en múltiples computadoras que operan en paralelo.

En algunas realizaciones, cuando el diagrama de salto se utiliza para modelar conjuntos ordenados de palabras bajo el marco de PENN, cada partición de clasificador y sus particiones de incrustación asociadas se pueden entrenar en paralelo completo para alcanzar el mismo estado que si no estuvieran distribuidas. En algunas realizaciones, el entrenamiento se puede lograr por múltiples computadoras sin intercomunicación en absoluto. En algunas realizaciones, la configuración de incrustación con ventanas puede entrenar todas las posiciones de ventana en paralelo completo y concatenar incrustaciones y clasificadores al final del entrenamiento. Dada la máquina *j,* se optimiza la siguiente función objetivo:

$$arg \quad max_{\theta}(\prod_{(w,C)\in d} p(c_j^i;\theta) \qquad \prod_{(w,C)\in d'} p(c_j^i;\theta))$$

donde c_j^i es la representación específica de la ubicación (partición *j)* para la palabra en la posición de la ventana *j* con relación a la palabra de enfoque *w.*

La concatenación de las matrices de peso syn_0 y syn_1 luego incorpora la suma sobre *j* de nuevo en la función objetivo del diagrama de salto de PENN durante el proceso de propagación hacia adelante, produciendo resultados de entrenamiento idénticos a los de una red entrenada de una manera de skip-gram de PENN de un solo subproceso y un solo modelo. Este estilo de entrenamiento logra resultados de entrenamiento de paridad con los métodos actuales de vanguardia en tanto que se entrena en paralelo en tantas *j* máquinas separadas como sea posible.

DIEM

En algunas realizaciones, las tareas sintácticas se pueden realizar por un proceso llamado incrustación por interpolación densa (DIEM). En esta técnica, los vectores característicos para unidades lingüísticas compuestas más grandes (tal como las palabras) se pueden calcular a partir de incrustaciones generadas en unidades lingüísticas más pequeñas (tal como las letras). Debido a que la estructura de palabras o frases se correlaciona más con la sintaxis que con el significado semántico, esta técnica funciona mejor en las tareas de analogía sintáctica. Para realizar este método, los vectores característicos se calculan primero para las unidades lingüísticas más pequeñas utilizando una red neuronal de acuerdo con una realización. Un vector característico para una unidad lingüística más grande se puede calcular al interpolar las incrustaciones para la unidad lingüística más pequeña sobre la unidad lingüística más grande. Por ejemplo, donde la unidad lingüística más pequeña es una letra, y la unidad lingüística más grande es una palabra, los vectores característicos para las palabras se pueden generar de acuerdo con el siguiente pseudocódigo:

ENTRADA: longitud de palabra *l,* listar incrustaciones de carácter (por ejemplo, la palabra) $char_t$, múltiplo *M,* dimensionalidad de carácter *C,* vector v_m.

```
for i = 0 to l 1 do
        s = M* i / l
        for m = 0 to M- 1 do
                d = pow(1 - (abs(s - m))/ M, 2)
                vm = vm + d* char_i
        end for
end for
```

El tamaño de incrustación final se puede seleccionar como un múltiplo *M* de la dimensionalidad de incrustación de carácter *C,* tal que la incrustación final contenga *M* sector de longitud *C.* Dentro de cada sector de la incrustación final, la incrustación de cada carácter se suma de acuerdo con un promedio ponderado. Este promedio ponderado se determina por la variable *d* anterior. Esta variable compara la distancia porcentual al cuadrado entre la posición del carácter *i* en la palabra de longitud *l* y la posición del sector *m* en la secuencia de *M* sectores. Esto se calcula tal que el carácter del extremo izquierdo esté a una distancia aproximada del 0% del sector de extremo izquierdo, y el carácter del extremo derecho esté a una distancia aproximada del 100% del sector de extremo derecho. 1 menos esta distancia, al cuadrado, se utiliza entonces para ponderar la incrustación de cada respectivo carácter en cada respectivo sector. Por ejemplo, el porcentaje de distancia a través de una palabra de longitud 4 es 0%, 33%, 66% y 100% (de izquierda a derecha a través de la palabra). La distancia porcentual a través de una secuencia de sectores de longitud 3 es 0%, 50% y 100%. Por lo tanto, la distancia entre el carácter 1 y el sector 3 es (0% - 100%). Al cuadrado, esto equivale al 100%. Uno menos este cuadrado es 0%. Por lo tanto, el carácter 1 en el sector 3 tendría un peso de 0% (nada). Lo contrario sería cierto para el carácter 4 y el sector 3, ya que ambos yacen en la posición extrema derecha en sus respectivas secuencias.

Para cada carácter de una palabra, su índice *i* se escala primero linealmente con el tamaño de la incrustación "sintáctica" final tal que *s = M * i /*l. Entonces, para cada posición *C* de longitud *m* (de *M* posiciones/sectores) en la incrustación de la palabra final *vm,* se calcula una distancia al cuadrado con relación al índice escalado tal que la distancia *d* = pow(1- (abs(s - j)) / M,2). El vector de carácter para el carácter en la posición *i* en la palabra

entonces se escala por *d* y se adiciona por elementos en la posición *j* del vector v.

En algunas realizaciones, un método de incrustación de interpolación densa se puede lograr de manera más eficiente al almacenar en memoria intermedia un conjunto de matrices de transformación, que son valores almacenados en memoria intermedia de d_{ij} para palabras de tamaño variable. Estas matrices se pueden utilizar para transformar vectores de caracteres concatenados de longitud variable en incrustaciones de palabras de longitud fija mediante multiplicación vector-matriz.

Los vectores sintácticos de acuerdo con algunas realizaciones también proporcionan ventajas de escalamiento y generalización significativas sobre los vectores semánticos. Por ejemplo, se pueden generar fácilmente nuevos vectores sintácticos para palabras nunca antes vistas, dando una generalización sin pérdidas a cualquier palabra del entrenamiento inicial de caracteres, asumiendo sólo que la palabra se compone de caracteres que se han visto. Las incrustaciones sintácticas se pueden generar de una manera completamente distribuida y sólo requieren una concatenación vectorial pequeña y una multiplicación vector-matriz por palabra. Además, en algunas realizaciones, los vectores de caracteres (habitualmente longitud 32) y las matrices de transformación (como mucho 20 o más de ellos) se pueden almacenar de manera muy eficiente con relación a los vocabularios semánticos, que pueden ser varios millones de vectores de dimensionalidad 1000 o más. En algunas realizaciones, DIEM funciona de manera óptima usando más de 6 órdenes de magnitud menos de espacio de almacenamiento, y más de 5 órdenes de magnitud menos de ejemplos de entrenamiento que las incrustaciones semánticas a nivel de palabra.

El marco de PENN modela la probabilidad de que se presente una palabra dadas las palabras que la circundan. Cambiar la estrategia de partición modifica esta distribución de probabilidad al capturar (o ignorar) varias sinergias. El modelado de cada permutación de la posición de la ventana alrededor de una palabra de enfoque se aproxima a la distribución completa, o la probabilidad condicional completa de que la palabra de enfoque se presente dada la posición y el valor de todas y cada una de las palabras en el contexto que la circunda.

El modelado de cada distribución permite que se capturen todas las sinergias potenciales en la incrustación combinada. El número de permutaciones posibles en esta estrategia de partición puede exceder fácilmente varios cientos dado un tamaño de ventana de 10 o más con cada modelo entrenado que contiene miles de millones de parámetros. Sin embargo, las estrategias de partición de palabras similares producen incrustaciones de palabras similares y, por extensión, una calidad de analogía de palabras similar en cada subtarea. Por lo tanto, algunas realizaciones pueden aproximarse a las perspectivas variables generadas utilizando la distribución de co-ocurrencia completa con menos modelos entrenados entrenando modelos suficientemente diferentes y concatenándolos. Este método utiliza diferentes distribuciones de probabilidad para generar incrustaciones de palabras que modelan diferentes perspectivas en cada palabra del vocabulario. Los vectores derivados de caracteres también incorporan una perspectiva única al capturar información de los caracteres de una palabra en lugar de los contextos en los que se presenta una palabra. Debido a que la incrustación representa relaciones sintácticas entre palabras desde una perspectiva única, el método de concatenación de acuerdo con las realizaciones descritas anteriormente se puede generalizar para incluir los vectores de DIEM.

Tareas de analogía

Un aspecto relacionado de las redes neuronales de acuerdo con algunas realizaciones es que la matriz de incrustación Syn0 se puede usar para tareas de analogía. De la misma manera que cada fila puede representar todas las entradas a una neurona específica, cada fila representa la "incrustación" característica de una palabra específica en la red neuronal. La incrustación de una palabra puede ser un vector que corresponde a todos o un subconjunto de los pesos de sinapsis para una partición individual, o todos o un subconjunto de todas las particiones, o variaciones de las mismas.

Estas incrustaciones de palabras características se pueden utilizar para realizar tareas de analogía. Por ejemplo, la analogía "rey es a la reina como el hombre es a la mujer" se puede codificar como rey - reina = hombre - mujer. Es decir, cada palabra se puede representar como un vector característico de incrustaciones, y cada relación como una diferencia entre los dos vectores característicos. Usando el ejemplo de rey/reina, una red neuronal de acuerdo con una realización se puede utilizar para derivar la respuesta "mujer" a la pregunta "rey es a la reina como el hombre es a qué?" (rey - reina = hombre - ?). El problema se puede resolver por la redisposición de las operaciones al hombre - rey + reina = ?. Esto da por resultado un vector de solución que entonces puede convertirse en una palabra de respuesta al comparar el vector de solución con el vector de incrustación para cada palabra en el diccionario usando similitud de coseno. Es decir:

$$\textit{similitud} = cos\,(\theta) = \frac{\textit{solución} \cdot \textit{palabra}}{\|\textit{solución}\| \, \|\textit{palabra}\|} = \frac{\sum_{i=1}^{n} \textit{solución}_i \textit{palabra}_i}{\sqrt{\sum_{i=1}^{n} \textit{solución}_i^2} \sqrt{\sum_{i=1}^{n} \textit{palabra}_i^2}}$$

Al buscar los vectores característicos de las palabras en el diccionario, y encontrar una palabra con la distancia de coseno más pequeña (y, por lo tanto, el más alto grado de similitud), usualmente dará por resultado encontrar la

respuesta "mujer". Dependiendo de la estructura de la red, el grado de entrenamiento, se pueden presentar otros resultados, como "niña" o "mujer", sin embargo, "mujer" casi siempre se encuentra entre las palabras más similares.

El ejemplo anterior dado es una tarea semántica, sin embargo, también se pueden realizar analogías sintácticas. Por ejemplo, la pregunta "¿correr es a correr como la poda es a?" se puede realizar de manera similar con una realización de la divulgación. Entre los mejores resultados habrá "poda", que comparte una sintaxis común, aunque con un significado semántico muy diferente.

En algunas realizaciones, las incrustaciones de palabras aprendidas a través de modelos de lenguaje neuronal pueden compartir recursos entre múltiples idiomas. Mediante el entrenamiento previo en ambos idiomas, las relaciones semánticas entre los idiomas se pueden codificar en las incrustaciones. En algunas realizaciones, por ejemplo, la aritmética vectorial simple de palabras de parada entre inglés y español puede aproximarse a la traducción automática:

(“film”) - (“is”) + (“es”) = (“película”)

Esta operación vectorial resta una palabra de parada en inglés ("is") de un sustantivo en inglés y adiciona la palabra de parada en español equivalente ("es") a la incrustación. La ecuación retorna la palabra española para película (“película”). Esto sugiere que el entrenamiento previo de lenguaje neuronal mapea las palabras en el espacio vectorial tal que el idioma se convierta en otra dimensión que el modelo de representaciones distribuidas. En algunas realizaciones, esta propiedad de las incrustaciones permite que un modelo de sentimiento entrenado en un idioma prediga en otro. En algunas realizaciones, un sentimiento entrenado en un idioma predecirá mejor en otro donde hay un vocabulario compartido entre idiomas. En algunas realizaciones, durante el entrenamiento previo, los modelos de lenguaje neuronal pueden construir una capa oculta usando neuronas muestreadas de acuerdo con las frecuencias de vocabulario. De esta manera, la construcción de capa oculta para cada idioma será similar a pesar de tener representaciones de capa de entrada completamente diferentes. La consistencia en la capa oculta puede permitir la alineación de conceptos semánticos en el espacio vectorial. Esta alineación puede permitir la predicción de sentimientos en todos los idiomas.

Implementaciones y resultados

A continuación, se describen implementaciones de diversos aspectos de la divulgación y los resultados correspondientes. Algunos datos experimentales se presentan en la presente con propósitos de ilustración y no se deben interpretar como que limitan el alcance de la presente divulgación de ninguna manera o que excluyen cualquier realización alternativa o adicional.

Los presentes inventores realizaron experimentos en tareas de analogía de palabras compuestas de una variedad de tareas de similitud de palabras. En concreto, se utilizó el conjunto de datos de analogía de Google, que contiene 19.544 preguntas, divididas en secciones semánticas y sintácticas. Ambas secciones se dividen adicionalmente en subcategorías con base en el tipo de analogía. Cada tarea de analogía se expresa como una pregunta "¿A es a B como C es a ?".

Todo el entrenamiento se realizó sobre el conjunto de datos disponible en el sitio web de Google word2vec (https://code.google.com/p/word2vec), utilizando la secuencia de comandos de evaluación de analogía de palabras empaquetada. El conjunto de datos contiene aproximadamente 8 mil millones de palabras recopiladas de English News Crawl, 1-Billion-Word Benchmark, UMBC Webbase y English Wikipedia. El conjunto de datos utilizado aprovecha la normalización data-phrase2.txt predeterminada en todo el entrenamiento, que incluye tanto tokens individuales como frases. Salvo que se especifique lo contrario, todos los parámetros para el entrenamiento y la evaluación son idénticos a los parámetros predeterminados especificados en el modelo grande predeterminado de word2vec.

La figura 6 muestra la exactitud relativa de cada partición en un modelo de PENN a juzgar por las puntuaciones de analogía de palabras relativas a filas. Otros experimentos indicaron que el patrón presente en el mapa de calor es consistente a través de los ajustes de parámetros. Hay una clara diferencia de calidad entre las posiciones de las ventanas que predicen hacia adelante (lado izquierdo de la figura) y las posiciones de las ventanas que predicen hacia atrás (lado derecho de la figura). "Moneda" logra la mayor parte de su poder predictivo en predicciones de corto alcance, en tanto que "países comunes de capital" es un gradiente mucho más suave sobre la ventana.

Estos patrones respaldan la intuición de que las diferentes posiciones de las ventanas desempeñan diferentes funciones en diferentes tareas.

Tabla A

Estilo de configuración	W2V	D	W & D
Estilo de entrenamiento	CBOW	CLOW	CLOW
Tamaño de vector de palabra	500	500	500
Tamaño de partición	500	250	250
Tamaño de ventana	10	10	1
capital-común	89,72	92,29	94,86
capital-mundo	92,11	92,46	90,96
moneda	14,63	19,95	12,37
ciudad en estado	78,76	72,48	69,56
familia	82,81	86,76	85,18
Total semántico	81,02	80,19	78,07
adjetivo a adverbio	37,70	35,08	35,08
opuesto	36,21	40,15	37,93
comparativo	86,71	87,31	93,39
superlativo	80,12	82,00	87,25
participio de presente	77,27	80,78	83,05
adjetivo de nacionalidad	90,43	90,18	88,49
tiempo pasado	72,37	73,40	75,90
plural	80,18	81,83	74,55
verbos plurales	58,51	63,68	78,97
Total sintáctico	72,04	73,45	74,59
Total combinado	76,08	76,49	76,16

La tabla A muestra el rendimiento de la implementación de CBOW predeterminada de word2vec con relación a la configuración direccional. Lo más destacado es que PENN supera la implementación de word2vec utilizando sólo un tamaño de ventana de 1, en tanto que word2vec se parametrizó con el valor predeterminado de 10. Además, se puede observar que el incremento de la dimensionalidad de word2vec de CBOW de punto de referencia pasado 500 logra un rendimiento subóptimo. Por lo tanto, una comparación justa de dos modelos debe estar entre la parametrización óptima (en lugar de igual) para cada modelo. Esto es especialmente importante puesto que los modelos de PENN están modelando una distribución de probabilidad mucho más rica, puesto que se está conservando el orden. Por lo tanto, los ajustes óptimos de los parámetros a menudo requieren una mayor dimensionalidad. Adicionalmente, en esta tabla, se puede observar que, a diferencia del word2vec de CBOW original, un tamaño de ventana más grande no siempre es mejor. Las ventanas más grandes tienden a crear incrustaciones un poco más semánticas, en tanto que los tamaños de ventana más pequeños tienden a crear incrustaciones un poco más sintácticas. Esto sigue la intuición de que la sintaxis desempeña una función importante en la gramática, que está dictada por reglas sobre qué palabras tienen sentido que se presenten inmediatamente una al lado de la otra. Las palabras que están separadas por +5 palabras se agrupan con base en la presente materia y la semántica en lugar de la gramática. Con respecto al tamaño de ventana y la calidad general, debido a que las particiones dividen el vector global para una palabra, el aumento del tamaño de ventana disminuye el tamaño de cada partición en la ventana si el tamaño de vector global permanece constante. Puesto que cada incrustación está intentando modelar una distribución de probabilidad muy compleja (cientos de miles de palabras), el tamaño de la partición en cada partición debe permanecer lo suficientemente alto para modelar esta distribución. Por lo tanto, el modelado de ventanas grandes para incrustaciones semánticas es óptimo cuando se utiliza el modelo de incrustación direccional, que tiene un tamaño de partición fijo de 2, o un tamaño de vector global grande. El modelo direccional con parámetros óptimos tiene una calidad ligeramente menor que el modelo de ventana con parámetros óptimos debido a que el promedio vectorial se presenta en cada panel de ventana.

Tabla B

Semántico		Sintáctico	
“general” - similitud			
secretario	0,619	gneral	0,986
elecciones	0,563	genral	0,978
motores	0,535	generalmente	0,954

subsecretario	0,534	generación	0,944
“ve” - “ver” + “banco” ●			
pedernal	0,580	bancos	0,970
yarda	0,545	banco	0,939
peres	0,506	vigas	0,914
c.c	0,500	prohibiciones	0,895

La tabla B documenta el cambio en la calidad de la consulta de analogía sintáctica como resultado de los vectores de DIEM interpolados. Para el experimento de DIEM, cada consultade analogía se realizó primero ejecutando la consulta en CLOW y DIEM de manera independiente, y seleccionando las mil principales similitudes de coseno de CLOW. Los presentes inventores sumaron la similitud de coseno al cuadrado de cada uno de estos mil principales con cada similitud de coseno asociada retornada por la DIEM, y recurrieron. Se encontró que esta era una estimación eficiente de la concatenación que no reducía la calidad.

Los resultados finales muestran un aumento en la calidad y el tamaño con respecto a los modelos anteriores, con un aumento sintáctico del 40% con respecto a otros resultados. Dentro de los modelos de PENN, existe una compensación entre la velocidad y el rendimiento entre SG-DIEM y CLOW-DIEM. A diferencia de word2vec, donde SG fue más lento y de mayor calidad, CLOW es el modelo más lento y logra una puntuación más alta. En este caso, los presentes inventores logran un nivel 20x de paralelismo en SG-DIEM con relación a CLOW, con cada partición de entrenamiento de modelo de 250 dimensiones (250 * 20 = 5000 dimensionalidad final). Una red de parámetros de 160 mil millones también se entrenó durante la noche en 3 CPU de múltiples núcleos, sin embargo, produjo vectores dimensionales de 20000 para cada palabra y posteriormente sobreajustó los datos de entrenamiento. Esto es debido a que un conjunto de datos de 8 mil millones de tokens con un parámetro de muestreo negativo de 10 tiene 80 mil millones de ejemplos de entrenamiento. Tener más parámetros que ejemplos de entrenamiento sobreajusta un conjunto de datos, mientras que 40 mil millones se desempeñan a la par con el estado actual de la técnica.

La figura 6 muestra la exactitud relativa de cada partición en un modelo de PENN a juzgar por las puntuaciones de analogía de palabras relativas a filas. Otros experimentos indicaron que el patrón presente en el mapa de calor es consistente a través de los ajustes de parámetros. Hay una clara diferencia de calidad entre las posiciones de las ventanas que predicen hacia adelante (lado izquierdo de la figura) y las posiciones de las ventanas que predicen hacia atrás (lado derecho de la figura). "Moneda" logra la mayor parte de su poder predictivo en predicciones de corto alcance, en tanto que "países comunes de capital" es un gradiente mucho más suave sobre la ventana. Estos patrones respaldan la intuición de que las diferentes posiciones de las ventanas desempeñan diferentes funciones en diferentes tareas.

Tabla C

Arquitectura semántica	CBOW	CLOW	DIEM
Dim. de Vector semántico	500	500	500
Total semántico	81,02	80,19	80,19
adjetivo a adverbio	37,70	35,08	94,55
opuesto	36,21	40,15	74,60
comparativo	86,71	87,31	92,49
superlativo	80,12	82,00	87,61
participio de presente	77,27	80,78	93,27
adjetivo de nacionalidad	90,43	90,18	71,04
tiempo pasado	72,37	73,40	47,56
plural	80,18	81,83	93,69
verbos plurales	58,51	63,68	95,97
Total sintáctico	72,04	73,45	81,53
Puntuación combinada	76,08	76,49	80,93

Tabla D

Estilo de configuración	W2V	D	W & D
Estilo de entrenamiento	CBOW	CLOW	CLOW
Tamaño de vector de palabra	500	500	500
Tamaño de partición	500	250	250
Tamaño de ventana	10	10	1
capital-común	89,72	92,29	94,86
capital-mundo	92,11	92,46	90,96
moneda	14,63	19,95	12,37
ciudad en estado	78,76	72,48	69,56
familia	82,81	86,76	85,18
Total semántico	81,02	80,19	78,07
adjetivo a adverbio	37,70	35,08	35,08
Opuesto	36,21	40,15	37,93
Comparativo	86,71	87,31	93,39
Superlativo	80,12	82,00	87,25
participio de presente	77,27	80,78	83,05
adjetivo de nacionalidad	90,43	90,18	88,49
tiempo pasado	72,37	73,40	75,90
Plural	80,18	81,83	74,55
verbos plurales	58,51	63,68	78,97
Total sintáctico	72,04	73,45	74,59
Total combinado	76,08	76,49	76,16

Las tablas C y D muestran el rendimiento de la implementación de CBOW predeterminada de word2vec con relación a la configuración direccional. Lo más destacado es que PENN supera la implementación de word2vec utilizando sólo un tamaño de ventana de 1, en tanto que word2vec se parametrizó con el valor predeterminado de 10. Además, se puede observar que el incremento de la dimensionalidad de word2vec de CBOW de punto de referencia pasado 500 logra un rendimiento subóptimo. Por lo tanto, una comparación justa de dos modelos debe estar entre la parametrización óptima (en lugar de igual) para cada modelo. Esto es especialmente importante puesto que los modelos de PENN están modelando una distribución de probabilidad mucho más rica, puesto que se está conservando el orden. Por lo tanto, los ajustes óptimos de los parámetros a menudo requieren una mayor dimensionalidad. Adicionalmente, en esta tabla, se puede observar que, a diferencia del word2vec de CBOW original, un tamaño de ventana más grande no siempre es mejor. Las ventanas más grandes tienden a crear incrustaciones un poco más semánticas, en tanto que los tamaños de ventana más pequeños tienden a crear incrustaciones un poco más sintácticas. Esto sigue la intuición de que la sintaxis desempeña una función importante en la gramática, que está dictada por reglas sobre qué palabras tiene sentido que se presenten inmediatamente una al lado de la otra. Las palabras que están separadas por +5 palabras se agrupan con base en la presente materia y la semántica en lugar de la gramática. Con respecto al tamaño de ventana y la calidad general, debido a que las particiones dividen el vector global para una palabra, el aumento del tamaño de ventana disminuye el tamaño de cada partición en la ventana si el tamaño de vector global permanece constante. Puesto que cada incrustación está intentando modelar una distribución de probabilidad muy compleja (cientos de miles de palabras), el tamaño de la partición en cada partición debe permanecer lo suficientemente alto para modelar esta distribución. Por lo tanto, el modelado de ventanas grandes para incrustaciones semánticas es óptimo cuando se utiliza el modelo de incrustación direccional, que tiene un tamaño de partición fijo de 2, o un tamaño de vector global grande. El modelo direccional con parámetros óptimos tiene una calidad ligeramente menor que el modelo de ventana con parámetros óptimos debido a que el promedio vectorial se presenta en cada panel de ventana.

La tabla B documenta el cambio en la calidad de la consulta de analogía sintáctica como resultado de los vectores de DIEM interpolados. Para el experimento de DIEM, cada consulta de analogía se realizó primero al ejecutar la consulta en CLOW y DIEM de manera independiente, y seleccionando las mil principales similitudes de coseno de CLOW. Los presentes inventores sumaron la similitud de coseno al cuadrado de cada uno de estos mil principales con cada similitud de coseno asociada retornada por la DIEM, y recurrieron. Se encontró que esta era una estimación eficiente de la concatenación que no reducía la calidad. Los resultados finales muestran un aumento en la calidad y el tamaño con respecto a los modelos anteriores con un aumento sintáctico del 40% sobre el mejor resultado publicado. Dentro de los modelos de PENN, existe una compensación entre la velocidad y el rendimiento

entre SG-DIEM y CLOW-DIEM. A diferencia de word2vec, donde SG fue más lento y de mayor calidad, CLOW es el modelo más lento y logra una puntuación más alta. En este caso, los presentes inventores lograron un nivel 20x de paralelismo en SG-DIEM con relación a CLOW, con cada partición de entrenamiento de modelo de 250 dimensiones (250 * 20 = 5000 dimensionalidad final). Una red de parámetros de 160 mil millones también se entrenó durante la noche en 3 CPU de múltiples núcleos, sin embargo, produjo vectores dimensionales de 20000 para cada palabra y posteriormente sobreajustó los datos de entrenamiento. Esto es debido a que un conjunto de datos de 8 mil millones de tokens con un parámetro de muestreo negativo de 10 tiene 80 mil millones de ejemplos de entrenamiento. Tener más parámetros que ejemplos de entrenamiento sobreajusta un conjunto de datos, en tanto que 40 mil millones se desempeñan a la par con el estado actual de la técnica, como se muestra en la tabla E.

Tabla E

Algoritmo	GloVe	Word2Vee		PENN+DIEM	
Config	X	CBOW	SG	SG	CLOW
Params	X	3,8 B	3,8 B	40B	16B
Sem. Dims	300	500	500	5000	2000
Semántico	81,9	81,0	82,2	69,6	82,3
Sintáctico	69,3	72,0	71,3	80,0	81,5
Combinado	75,0	76,1	76,2	75,3	80,9

Los presentes inventores realizaron experimentos adicionales para aproximar la distribución condicional completa de incrustaciones de palabras. Estos experimentos también se realizaron en tareas de analogía de palabras en el conjunto de datos de analogía de Google. Este conjunto de datos contiene 19.544 preguntas que preguntan "¿a es a b como c es a ?" y se divide en 14 subcategorías, 5 semánticas y 9 sintácticas.

El entrenamiento se realiza sobre el conjunto de datos 'data-phrase2.txt' disponible en el sitio web de Google word2vec utilizando la secuencia de comandos de evaluación de analogía de palabras empaquetada para consultar cada modelo individual. Debido a que consultar cada modelo semántico requiere hasta 128 GB de RAM, se utilizó una aproximación de concatenación normalizada similar a esa de los experimentos descritos anteriormente. Cada analogía se consultó contra cada modelo semántico en paralelo, y se guardaron los 1000 mejores resultados. La puntuación de cada palabra se sumó en los diversos modelos para crear una puntuación global para cada palabra. La palabra con el valor máximo se seleccionó como la predicción del conjunto. En los experimentos de los presentes inventores, cada puntuación en cada modelo semántico (modelo de PENN) se elevó a la potencia de 0,1 antes de sumarse. Se encontró que esto normalizaba los modelos a intervalos de confianza similares.

Para las consultas de analogía sintáctica, las 100 palabras principales del paso semántico se seleccionaron para crear incrustaciones sintácticas. La consulta de analogía se realizó utilizando las incrustaciones sintácticas tal que se generaran puntuaciones de distancia de coseno basadas en incrustaciones sintácticas. Estas puntuaciones se sumaron por elementos con puntuaciones semánticas en las 100 palabras principales originales. Entonces se seleccionó la palabra principal como la respuesta de analogía final. Este paso se omitió por analogías provenientes de categorías semánticas. Las puntuaciones sintácticas se normalizaron al elevarlas a la potencia de 10.

El experimento descrito a continuación se aproxima a la distribución condicional completa al concatenar incrustaciones de una variedad de configuraciones. Estas configuraciones se muestran en la tabla F a continuación:

Tabla F

Estilo de entrenamiento	Tamaño de ventana	Dimensionalidad
WENN	10	500
DENN	5	500
WENN	2	2000
DENN	5	2000
DENN	10	2000
DENN	1	500
DIEM	X	320

Los resultados de estos experimentos se muestran en la tabla G:

Tabla G

	SG	PENN	Completo
capital-común	94,47	92,29	95,65
capital-mundo	94,08	92,46	93,90
moneda	15,82	19,95	17,32
ciudad en estado	81,15	72,48	78,88
familia	67,00	86,76	85,38
Semántico	82,17	80,19	82,70
adjetivo a adverbio	41,53	94,55	90,73
opuesto	37,32	74,60	73,15
comparativo	86,04	92,49	99,70
superlativo	71,21	87,61	91,89
participio de presente	76,61	93,27	93,66
adjetivo de nacionalidad	92,31	71,04	91,43
tiempo pasado	64,81	47,56	60,01
plural	86,11	93,69	97,90
verbos plurales	58,85	95,97	95,86
Sintáctico	71,33	81,53	88,29
Total	76,22	80,93	85,77

Se debe observar que adicionar modelos adicionales al conjunto incrementó la puntuación general, incluso cuando el modelo que se agregó ya tenía una puntuación más baja que el conjunto. A nivel granular, esto es el resultado de la singularidad del modelo que se adiciona. Las fallas de analogía de palabras que produce el modelo son significativamente diferentes de las fallas que produce el conjunto, tal que su combinación logre una puntuación más alta que cualquiera de las dos.

Los resultados finales muestran un aumento en la calidad con respecto a los modelos anteriores, con una reducción de errores del 59,2% en las puntuaciones sintácticas y una reducción general de errores del 40,2% con relación a word2vec. La entrenamiento en incrustaciones semánticas de palabras con base en distribuciones de probabilidad variables, normalización y concatenación es beneficiosa para las tareas de analogía de palabras, logrando un rendimiento de vanguardia en categorías semánticas y sintácticas por un margen significativo. Estos resultados son con base en la intuición de que diferentes distribuciones modelan diferentes perspectivas sobre la misma palabra, el agregado del cual es más expresivo que cada incrustación individualmente.

Los experimentos se realizaron adicionalmente utilizando realizaciones para realizar el modelado del lenguaje neuronal en diferentes idiomas, todos realizados con el corpus de revisión de películas de IMDB. El corpus consta de 100.000 revisiones de películas que están etiquetadas para la polaridad del sentimiento (positivo / negativo). Las revisiones se dividen en secciones de entrenamiento, pruebas y desarrollo. Los conjuntos de entrenamiento y pruebas tienen 25.000 revisiones cada uno con 12.500 revisiones positivas y 12.500 negativas. Los 50.000 documentos restantes comprenden el conjunto de desarrollo.

Los presentes inventores utilizaron la API de Google Translate para seleccionar un corpus de revisión de películas en español. Los sistemas tal como la API de traducción de Google se han evaluado por la investigación en la literatura de NLP y en general se aceptan como un mecanismo con el cual curar conjuntos de datos etiquetados en nuevos idiomas [1][2][3]. Para cada revisión en el corpus original de IMDB en inglés, se realizó una solicitud de API a la API de Google Translate para traducir el texto completo de la revisión al español. Las traducciones a nivel de párrafo resultantes mantuvieron la misma etiqueta de polaridad, así como el ID, a fin de mantener comparaciones individuales entre los corpus de español e inglés.

Los presentes inventores utilizaron la arquitectura de omisión de word2vec para aprender 100 representaciones de características dimensionales para palabras en el corpus por el uso de softmax jerárquico y muestreo negativo (con 10 muestras) para las representaciones de palabras de enfoque. El preprocesamiento consta de minúsculas, separar las palabras de la puntuación, y remover las etiquetas HTML. Después de aprender las representaciones de palabras, las palabras de parada se filtran utilizando el corpus de palabras de parada de NLTK [4] y las representaciones a nivel de revisión se crean al promediar las características de cada palabra restante en la revisión.

Los presentes inventores utilizaron la configuración de bolsa de palabras distribuida (dbow) de Paragraph Vector para aprender 100 representaciones de características dimensionales para revisiones por el uso de muestreo jerárquico softmax y negativo (con 10 muestras) para las representaciones de palabras de enfoque. El preprocesamiento consta de minúsculas, separar las palabras de la puntuación, y remover las etiquetas HTML. Las palabras de parada se conservan en las revisiones.

Los presentes inventores utilizaron la arquitectura de skip-gram de PENN para aprender 100 representaciones de características dimensionales para palabras en el corpus mediante el uso de muestreo negativo (con 10 muestras) y sin softmax jerárquico para las representaciones de enfoque. Se utilizaron particiones de posiciones de ventana

-9 a -4. Los pasos de preprocesamiento y promediado son idénticos al proceso descrito en la sección de configuración de word2vec.

El ensamblaje del modelo se realizó utilizando una implementación aumentada del trabajo reciente de [5]. Para cada corpus (inglés y español) se capacitó un modelo de palabras de validación y un modelo de palabras completo. Los modelos de palabras de validación (word2vec y PENN) se entrenan en 20.000 revisiones de los conjuntos de entrenamiento (10.000 positivas, 10.000 negativas) y 50.000 revisiones en los conjuntos de desarrollo. Las 5000 revisiones restantes en los conjuntos de entrenamiento se utilizan como conjuntos de validación. Los modelos de palabras completas se entrenan en todos los conjuntos de entrenamiento y conjuntos de desarrollo. Las representaciones vectoriales de párrafos se aprenden utilizando una entrenamiento previa no supervisada en todo el corpus (entrenamiento, desarrollo y pruebas) utilizando la implementación proporcionada con [5],

Modelos de un solo lenguaje entrenan previamente modelos de palabras en inglés y español de manera independiente entre sí. Las representaciones de las características en inglés y español se aprenden en modelos separados no supervisados. Sus evaluaciones actúan como un punto de referencia con el que interpretar los modelos multilingües en experimentos posteriores. Las representaciones a nivel de revisión se utilizan como características en una máquina de vector de soporte (SVM). Se utilizó un clasificador separado para cada idioma. Los modelos no supervisados para inglés y español nunca interactúan. Los modelos en español se evaluaron contra el conjunto de pruebas en español y los modelos en inglés contra el conjunto de pruebas en inglés. Las puntuaciones resultantes para estos modelos se proporcionan en la tabla H.

Tabla H

Entrenamiento previo	Formación	Idioma de destino	Modelo	Porcentaje de exactitud
Inglés	Inglés	Inglés	Word2Vec	88,42
			Paragraph Vector	88,55
			PENN	87,90
			Conjunto	89,22
Español	Español	Español	Word2Vec	87,31
			Paragraph Vector	85,3
			PENN	82,25
			Conjunto	86,63

Los modelos multilingües entrenan previamente utilizando inglés y español para aprender representaciones. Las representaciones de las características en inglés y español se aprenden en el mismo modelo no supervisado. Las representaciones a nivel de revisión se utilizan como características en un SVM. El clasificador está entrenado en ejemplos en inglés. No se realiza ningún entrenamiento sobre el corpus de español. Las funciones aprendidas en el entrenamiento previo multilingüe permiten que los recursos de funciones se compartan entre tanto español como inglés. Los modelos resultantes se evalúan en el corpus de español. Las puntuaciones para estos modelos se proporcionan en la tabla I.

Tabla I

Entrenamiento previo	Formación	Idioma de destino	Modelo	Porcentaje de exactitud
Inglés + Español	Inglés	Inglés	Word2Vec	77,54
			Paragraph Vector	86,02
			PENN	78,33
			Conjunto	85,11
Inglés + Español	Inglés	Español	Word2Vec	80,08
			Paragraph Vector	78,86
			PENN	77,54
			Conjunto	85,51

Los modelos de características bilingües ensamblan modelos de un solo lenguaje con modelos multilingües, los presentes inventores aprovechan las asignaciones de ID entre los corpus de inglés y español para ensamblar cada combinación de rutinas de entrenamiento previo y entrenamiento mencionadas en la sección anterior. Además, se incorporaron modelos de n-gramas (sin entrenamiento previo) que están entrenados en inglés. El primer modelo de n-gramas es un Perceptrón Promediado con características de n-gramas de alta afinidad (elegido a través de Person's como se propone en [6]). El segundo modelo de n-gramas es una SVM de Naive Bayes como se propone en [7]. Mantener ID entre los idiomas permite obtener múltiples predicciones para cada revisión: algunos modelos predicen en la traducción al inglés de la revisión, otros modelos predicen en la traducción al español de la revisión, creando una puntuación de sentimiento de revisión ensamblada. Este ensamblaje excede el estado de la técnica para la clasificación de polaridad de IMDB. Estas puntuaciones se proporcionan en la tabla J a continuación:

Tabla J

Entrenamiento previo	Formación	Idioma de destino	Modelo	Porcentaje de exactitud
Inglés	Inglés	Inglés	Word2Vec	88,42
			Paragraph Vector	88,55
			PENN	87,90
			Conjunto	89,22
Español	Español	Español	Word2Vec	87,31
			Paragraph Vector	85,30
			PENN	82,25
Ninguno	Inglés	Inglés	NBSVM-TRJ	91,87
			Perceptrón promediado	87,75
Conjunto completo				94,20
Estado de la técnica anterior				92,57

Los experimentos de los presentes inventores demuestran la eficacia del modelado de lenguaje neuronal para la clasificación de sentimientos multilingüe. Los modelos de lenguaje neuronal codifican relaciones significativas entre palabras al mapearlas al espacio vectorial, y esta propiedad de sus incrustaciones ayuda a explicar por qué los modelos multilingües pueden predecir de manera efectiva en idiomas en los que no se han entrenado. Los presentes inventores emplean tres técnicas diferentes para el entrenamiento previo no supervisado y muestran que cada técnica es capaz de codificar relaciones semánticas entre idiomas. Los presentes inventores entrenaron modelos de un solo lenguaje de punto de referencia con los que comparar las predicciones multilingüe (tabla H). La puntuación de referencia más alta para el español es del 87,31% (word2vec) y la puntuación de conjunto es del 86,63%. Usando las puntuaciones de referencia para la comparación, los presentes inventores evalúan modelos multilingües en el corpus de español. Sin entrenamiento en español, los modelos son capaces de clasificar la polaridad con un 80% de exactitud, y cuando estos mismos modelos se ensamblan juntos clasifican la polaridad con una exactitud del 85,51% (Tabla I). Por lo tanto, los presentes inventores logran un margen de 2% entre el punto de referencia, los modelos de un solo lenguaje y los modelos multilingües. La exactitud general de los modelos multilingües incrementa en un 5% cuando se ensamblan. Este comportamiento sugiere que el entrenamiento previo para cada idioma captura patrones y estructuras ligeramente diferentes que, en última instancia, ayudan al clasificador a predecir exactamente las etiquetas para aquellos ejemplos cerca del límite de decisión.

Además de los ejemplos descritos anteriormente para implementar diversos aspectos de la presente divulgación en diversos campos de aplicación, los aspectos de la presente divulgación se pueden implementar en numerosos otros campos, que incluyen, pero no se limitan a, atención médica, por ejemplo, en monitoreo clínico en tiempo real y generación de reportes históricos sobre poblaciones de pacientes. Algunas realizaciones descritas en la presente se pueden aplicar en apoyo de decisiones clínicas y otras tareas que requieren resúmenes de texto no estructurado. En un caso de uso de ejemplo, los aspectos de la presente divulgación se pueden utilizar para modelar la probabilidad de que un paciente desarrolle una infección, que incluye una infección sistémica de cuerpo completo. En términos más generales, los aspectos de la presente divulgación se pueden utilizar para predecir si un paciente desarrollará o no una condición de salud particular. Los registros de salud electrónicos (EHR) para varios pacientes se pueden usar con aspectos de la presente divulgación como se describe anteriormente para aprender representaciones de las posiciones de los ganglios y usar incrustaciones para predecir si un paciente desarrollara una infección. Las ubicaciones de grupos particulares en el espacio vectorial pueden ser representativas de estas probabilidades de desarrollar ciertas condiciones.

La figura 7 es un diagrama de arquitectura informática de un sistema informático. El sistema informático incluye una computadora 700 que se puede configurar para realizar una o más funciones asociadas con la presente tecnología divulgada. La computadora 700 incluye una unidad de procesamiento 702, una memoria de sistema 704, y un bus de sistema 706 que acopla la memoria 704 a la unidad de procesamiento 702. La computadora 700 incluye además un dispositivo de almacenamiento masivo 712 para almacenar módulos de programa 714. Los módulos de programa 714 pueden incluir módulos ejecutables por computadora para realizar la una o más funciones asociadas con las figuras 1-6. El dispositivo de almacenamiento masivo 712 incluye además un almacén de datos 716. El dispositivo de almacenamiento masivo 712 se conecta a la unidad de procesamiento 702 a través de un controlador de almacenamiento masivo (no mostrado) conectado al bus 706. El dispositivo de almacenamiento masivo 712 y sus medios de almacenamiento de computadora asociados proporcionan almacenamiento no volátil para la computadora 700. Aunque la descripción de medios de almacenamiento leíbles por computadora contenidos en la presente se refiere a un dispositivo de almacenamiento masivo, tal como un disco duro o unidad de CD-ROM, se debe apreciar por aquellos expertos en la técnica que los medios de almacenamiento leíbles por computadora pueden ser cualquier medio de almacenamiento leíble por computadora

disponible al que se puede acceder y leer por la computadora 700.

A manera de ejemplo, y sin limitación, los medios de almacenamiento leíbles por computadora pueden incluir medios volátiles y no volátiles, removibles y no removibles implementados en cualquier método o tecnología para el almacenamiento de información tal como instrucciones de almacenamiento de computadora, estructuras de datos, módulos de programa, u otros datos. Por ejemplo, los medios de almacenamiento leíbles por computadora incluyen, pero no se limitan a, RAM, ROM, EPROM, EEPROM, memoria flash u otra tecnología de memoria de estado sólido, CD-ROM, discos versátiles digitales ("DVD"), HD-DVD, BLU-RAY u otro almacenamiento óptico, casetes magnéticos, cinta magnética, almacenamiento de disco magnético u otros dispositivos de almacenamiento magnético, o cualquier otro medio que se pueda utilizar para almacenar la información deseada y al que se pueda acceder por la computadora 700. Los medios de almacenamiento leíbles por computadora como se describen en la presente no incluyen señales transitorias.

La computadora 700 puede operar en un entorno en red utilizando conexiones lógicas a computadoras remotas a través de una red 718. La computadora 700 se puede conectar a la red 718 a través de una unidad de interfaz de red 710 conectada al bus 706. Se debe apreciar que la unidad de interfaz de red 710 también se puede utilizar para conectarse a otros tipos de redes y sistemas de computadora remotos. La computadora 700 también puede incluir un controlador de entrada/salida 708 para recibir y procesar la entrada de una cantidad de dispositivos de entrada. Los dispositivos de entrada pueden incluir, pero no se limitan a, teclados, ratones, lápiz óptico, pantallas táctiles, micrófonos, dispositivos de captura de audio, o dispositivos de captura de imagen/vídeo. Un usuario final puede utilizar estos dispositivos de entrada para interactuar con una interfaz de usuario tal como una interfaz gráfica de usuario para gestionar diversas funciones realizadas por la computadora 700. El bus 706 puede permitir que la unidad de procesamiento 702 lea código y/o datos hacia/desde el dispositivo de almacenamiento masivo 712 u otros medios de almacenamiento de computadora. Los medios de almacenamiento leíbles por computadora pueden representar aparatos en forma de elementos de almacenamiento que se implementan usando cualquier tecnología adecuada, incluyendo, pero no limitado a semiconductores, materiales magnéticos, óptica, o similares. Los módulos de programa 714 incluyen instrucciones de software que, cuando se cargan en la unidad de procesamiento 702 y se ejecutan, provocan que la computadora 700 proporcione funciones para el modelado de lenguaje neuronal de acuerdo con aspectos de la presente divulgación descritos en la presente con referencia a realizaciones de ejemplo.

Los módulos de programa 714 también pueden proporcionar diversas herramientas o técnicas por las cuales la computadora 700 puede participar dentro de los sistemas generales o entornos operativos utilizando los componentes, flujos, y estructuras de datos analizados de principio a fin de esta descripción. En general, los módulos de programa 714 pueden, cuando se cargan en la unidad de procesamiento 702 y se ejecutan, transformar la unidad de procesamiento 702 y la computadora general 700 de un sistema informático de propósito general en un sistema informático de propósito especial. La unidad de procesamiento 702 se puede construir a partir de cualquier cantidad de transistores u otros elementos de circuito discretos, que pueden asumir individual o colectivamente cualquier cantidad de estados. Más específicamente, la unidad de procesamiento 702 puede funcionar como una máquina de estado finito, en respuesta a instrucciones ejecutables contenidas dentro de los módulos de programa 714. Estas instrucciones ejecutables por computadora pueden transformar la unidad de procesamiento 702 al especificar cómo la unidad de procesamiento 702 transita entre estados, transformando de esta manera los transistores u otros elementos de hardware discretos que constituyen la unidad de procesamiento 702.

La codificación de los módulos de programa 714 también puede transformar la estructura física de los medios de almacenamiento de computadora. La transformación específica de la estructura física puede depender de varios factores, en diferentes implementaciones de esta descripción. Los ejemplos de estos factores pueden incluir, pero no se limitan a: la tecnología utilizada para implementar los medios de almacenamiento de computadora, si los medios de almacenamiento de computadora se caracterizan como almacenamiento primario o secundario, y similares. Por ejemplo, si los medios de almacenamiento de computadora se implementan como memoria basada en semiconductores, el módulo de programa 714 puede transformar el estado físico de la memoria de semiconductores, cuando el software se codifica en el mismo. Por ejemplo, los módulos de programa 714 pueden transformar el estado de transistores, condensadores, u otros elementos de circuito discretos que constituyen la memoria de semiconductor.

Como otro ejemplo, el medio de almacenamiento leíble por computadora se puede implementar utilizando tecnología magnética u óptica. En estas implementaciones, los módulos de programa 714 pueden transformar el estado físico de los medios magnéticos u ópticos, cuando el software se codifica en los mismos. Estas transformaciones pueden incluir alterar las características magnéticas de ubicaciones particulares dentro de medios magnéticos determinados. Estas transformaciones también pueden incluir la alteración de las características físicas o características de ubicaciones particulares dentro de medios ópticos determinados, para cambiar las características ópticas de aquellas ubicaciones. Se debe apreciar que otras transformaciones de medios físicos son posibles sin desviarse del alcance de la presente divulgación.

Referencias

[1] Balahur, Alexandra, y Marco Turchi. “Improving Sentiment Analysis in Twitter Using Multilingual Machine Translated Data.” RANLP. 2013.

[2] Balahur, Alexandra, y Marco Turchi. ”Multilingual sentiment analysis using machine translation?.” Proceedings of the 3rd Workshop in Computational Approaches to Subjectivity and Sentiment Analysis. Association for Computational Linguistics, 2012.

[3] Wan, Xiaojun. “Using bilingual knowledge and ensemble techniques for unsupervised Chinese sentiment analysis.” Proceedings of the Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2008.

[4] Bird, Steven. ”NLTK: the natural language toolkit.” Proceedings of the COLING/ACL on Interactive presentation sessions. Association for Computational Linguistics, 2006.

[5] Mesnil, Gr’egoire, et al. “Ensemble of Generative and Discriminative Techniques for Sentiment Analysis of Movie Reviews.” arXiv preprint arXiv: 1412.5335 (2014).

[6] Manning, Christopher D. y Schutze, Hinrich. ”Foundations of Statistical Natural Language Processing.” 1999.

[7] Wang, Sida, y Christopher D. Manning. ”Baselines and bigrams: Simple, good sentiment and topic classification.” Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2. Association for Computational Linguistics, 2012.

REIVINDICACIONES

1. Un método implementado por computadora, que comprende:

recibir, en una pluralidad de nodos neuronales de entrada de una red neuronal implementada por computadora, una entrada que comprende una lista ordenada de una pluralidad de unidades lingüísticas con una unidad lingüística omitida, en donde cada uno de los nodos neuronales de entrada corresponde a una unidad lingüística seleccionada de la lista ordenada de una pluralidad de unidades lingüísticas y las unidades lingüísticas comprenden frases, palabras, letras u otros caracteres utilizados en el idioma;
codificar tipo *one-hot*, en la pluralidad de nodos neuronales de entrada, la entrada recibida por los nodos neuronales de entrada;
recibir, en una pluralidad de nodos neuronales en cada una de una pluralidad de particiones de nodo de incrustación en una capa de incrustación de la red neuronal implementada por computadora, una entrada de un conjunto separado de los nodos neuronales de entrada, y generar una salida al multiplicar al menos la entrada de cada nodo neuronal de entrada por uno de una pluralidad de pesos de entrada, en donde cada una de las particiones de nodo de incrustación corresponde a una posición en la lista ordenada con respecto a un término de enfoque y comprende una pluralidad de nodos neuronales, en donde el término de enfoque es una unidad lingüística omitida de la lista ordenada de la pluralidad de unidades lingüísticas; y
recibir, en cada nodo neuronal en una capa clasificadora de la red neuronal implementada por computadora, la salida de cada uno de los nodos neuronales de la capa de incrustación, para generar una salida al multiplicar al menos la salida de cada nodo neuronal de la capa de incrustación por uno de una pluralidad de pesos de entrada, y en donde la salida corresponde a una probabilidad de que una unidad lingüística particular sea el término de enfoque;
en donde cada nodo neuronal de una partición de la capa de incrustación se entrena independientemente de otras particiones; y
en donde la pluralidad de nodos neuronales de entrada, la capa de incrustación, al menos una capa oculta y la capa clasificadora forman la red neuronal y el método comprende además entrenar la red neuronal al:

entrenar una primera partición de las particiones de nodo de incrustación al:
remover el término de enfoque de la lista ordenada de unidades lingüísticas;
seleccionar una partición de las incrustaciones de cada unidad lingüística restante con base en la posición de esa respectiva unidad lingüística con relación al término de enfoque; y
actualizar los pesos para cada nodo neuronal en la capa clasificadora y la capa de incrustación tal que la probabilidad de la presencia del término de enfoque sea de aproximadamente el 100% y otras unidades lingüísticas muestreadas aleatoriamente sean del 0%; y entrenar una segunda partición al:
remover el término de enfoque de la lista ordenada de unidades lingüísticas;
seleccionar una partición de la incrustación de cada unidad lingüística restante con base en la posición de esa unidad lingüística con relación al término de enfoque; y
actualizar los pesos para modelar la probabilidad de la presencia del término de enfoque es de aproximadamente el 100% y otras unidades lingüísticas muestreadas aleatoriamente son del 0%,

en donde los pasos de entrenar una primera partición y entrenar una segunda partición se realizan en paralelo en dos computadoras diferentes.

2. El método implementado por computadora de la reivindicación 1, donde la unidad lingüística es un carácter o una palabra.

3. El método implementado por computadora de la reivindicación 1, donde las posiciones con relación a un término de enfoque de las particiones de nodo de incrustación son posiciones de ventana con relación al término de enfoque o direcciones con relación al término de enfoque.

4. El método implementado por computadora de la reivindicación 1, donde el método comprende además entrenar la red neuronal al realizar funciones que comprenden:
remover el término de enfoque de la lista ordenada de unidades lingüísticas;
seleccionar una partición de las incrustaciones de cada unidad lingüística restante con base en la posición de esa unidad lingüística con relación al término de enfoque;
concatenar las particiones;
propagar las particiones a través de la capa clasificadora; y
actualizar los pesos para uno o más nodos neuronales en la capa clasificadora y la capa de incrustación tal que la probabilidad de la presencia del término de enfoque sea de aproximadamente el 100% y otras unidades lingüísticas muestreadas aleatoriamente sean de aproximadamente el 0%.

5. El método implementado por computadora de la reivindicación 1, donde las unidades lingüísticas particulares se seleccionan de la lista ordenada de la pluralidad de unidades lingüísticas y múltiples dominios lingüísticos asociados con las unidades lingüísticas particulares seleccionadas se modelan en un espacio vectorial común, en donde cada uno de los múltiples dominios lingüísticos corresponde a un idioma diferente.

6. El método implementado por computadora de la reivindicación 1, donde los pesos de entrada para cada partición de nodo de incrustación se entrenan independientemente de las otras particiones de nodo de incrustación.

7. Un sistema que tiene dos computadoras configuradas para implementar:

una pluralidad de nodos neuronales de entrada, en donde la pluralidad de nodos neuronales de entrada se configura para:

recibir una entrada que comprende una lista ordenada de una pluralidad de unidades lingüísticas con una unidad lingüística omitida, en donde cada nodo neuronal de entrada de la pluralidad de nodos neuronales de entrada corresponde a una unidad lingüística seleccionada de la lista ordenada de una pluralidad de unidades lingüísticas y las unidades lingüísticas comprenden frases, palabras, letras u otros caracteres utilizados en el idioma; y
codificar tipo *one-hot* la entrada recibida por los nodos neuronales de entrada;
una capa de incrustación que comprende una pluralidad de particiones de nodo de incrustación, en donde cada una de las particiones de nodo de incrustación corresponde a una posición en la lista ordenada con relación a un término de enfoque y comprende una pluralidad de nodos neuronales, en donde el término de enfoque es una unidad lingüística omitida de la lista ordenada de la pluralidad de unidades lingüísticas, la pluralidad de nodos neuronales de cada una de las particiones de nodo de incrustación configurada para:
recibir una entrada de un conjunto separado de los nodos neuronales de entrada; y
calcular una salida multiplicando al menos la entrada de cada nodo neuronal de entrada por uno de una pluralidad de pesos de entrada; y

una capa clasificadora que comprende una pluralidad de nodos neuronales, cada nodo neuronal en la capa clasificadora configurada para:

recibir la salida de cada nodo neuronal de la capa de incrustación; y
generar una salida al multiplicar al menos la salida de cada nodo neuronal de la capa de incrustación por uno de una pluralidad de pesos de entrada, y en donde la salida corresponde a una probabilidad de que una unidad lingüística particular sea el término de enfoque;

en donde cada nodo neuronal de una partición de la capa de incrustación se entrena independientemente de otras particiones; y
en donde la pluralidad de nodos neuronales de entrada, la capa de incrustación, al menos una capa oculta y el clasificador forman una red neuronal, la red neuronal entrenada al:

entrenar una primera partición de las particiones de nodo de incrustación al:
remover el término de enfoque de la lista ordenada de unidades lingüísticas;
seleccionar una partición de las incrustaciones de cada unidad lingüística restante con base en la posición de esa respectiva unidad lingüística con relación al término de enfoque; y
actualizar los pesos para cada nodo neuronal en la capa clasificadora y la capa de incrustación tal que la probabilidad de la presencia del término de enfoque sea de aproximadamente el 100% y otras unidades lingüísticas muestreadas aleatoriamente sean del 0%; y
entrenar una segunda partición al:
remover el término de enfoque de la lista ordenada de unidades lingüísticas;
seleccionar una partición de la incrustación de cada unidad lingüística restante con base en la posición de esa unidad lingüística con relación al término de enfoque; y
actualizar los pesos para modelar la probabilidad de la presencia del término de enfoque es de aproximadamente el 100% y otras unidades lingüísticas muestreadas aleatoriamente son del 0%,

en donde los pasos de entrenar una primera partición y entrenar una segunda partición se realizan en paralelo en las dos computadoras.

8. El sistema de la reivindicación 7, donde la unidad lingüística es un carácter o una palabra.

9. El sistema de la reivindicación 7, donde las posiciones con relación a un término de enfoque de las particiones de nodo de incrustación son posiciones de ventana con relación al término de enfoque o direcciones con relación al término de enfoque.

10. El sistema de la reivindicación 7, donde las unidades lingüísticas particulares se seleccionan de la lista ordenada de la pluralidad de unidades lingüísticas y múltiples dominios lingüísticos asociados con las unidades lingüísticas particulares seleccionadas se modelan en un espacio vectorial común, en donde cada uno de los múltiples dominios lingüísticos corresponde a un idioma diferente.

11. El sistema de la reivindicación 7, donde los pesos de entrada para cada partición de nodo de incrustación se entrenan independientemente de las otras particiones de nodo de incrustación.

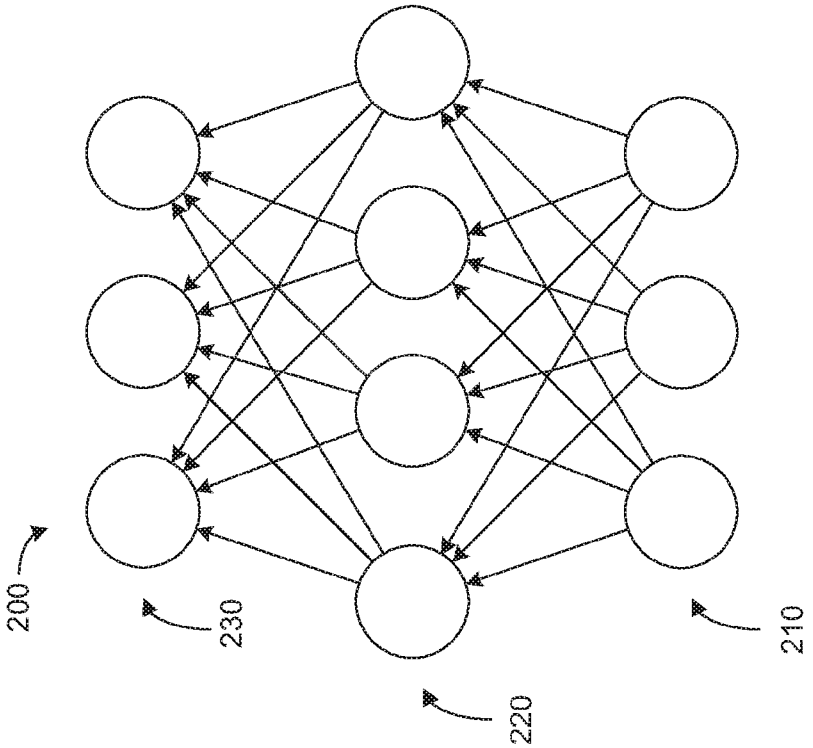
200
230
220
210

FIG. 2

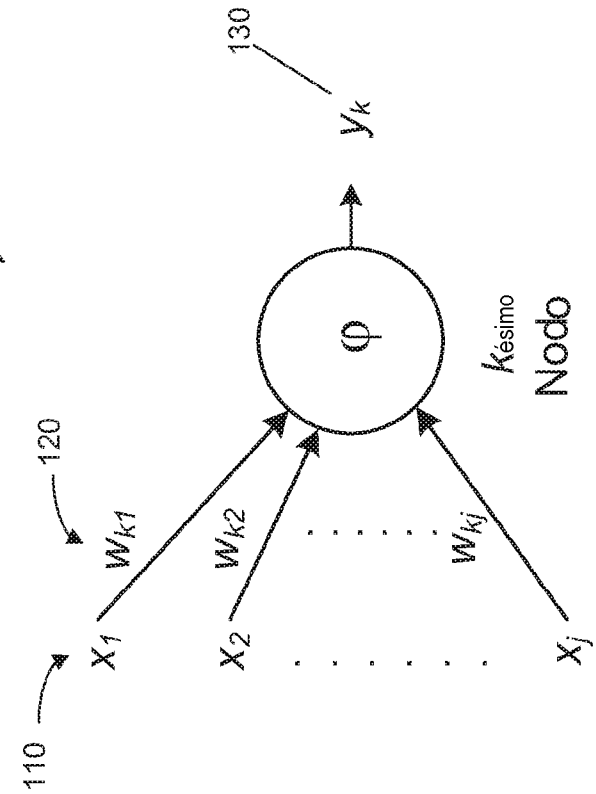
100
130
y_k
120
w_{k1}
w_{k2}
w_{kj}
φ
$k_{ésimo}$ Nodo
110
x_1
x_2
x_j

FIG. 1

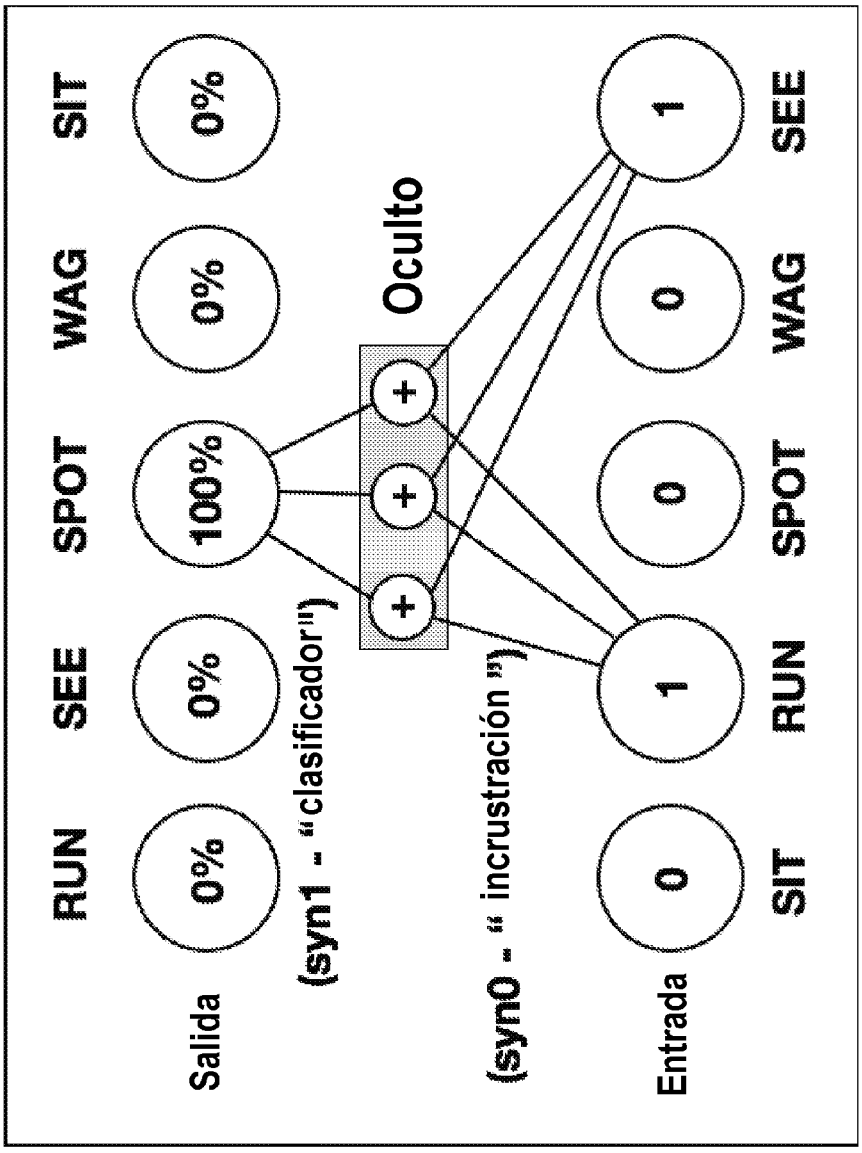

FIG. 3

TÉCNICA ANTERIOR

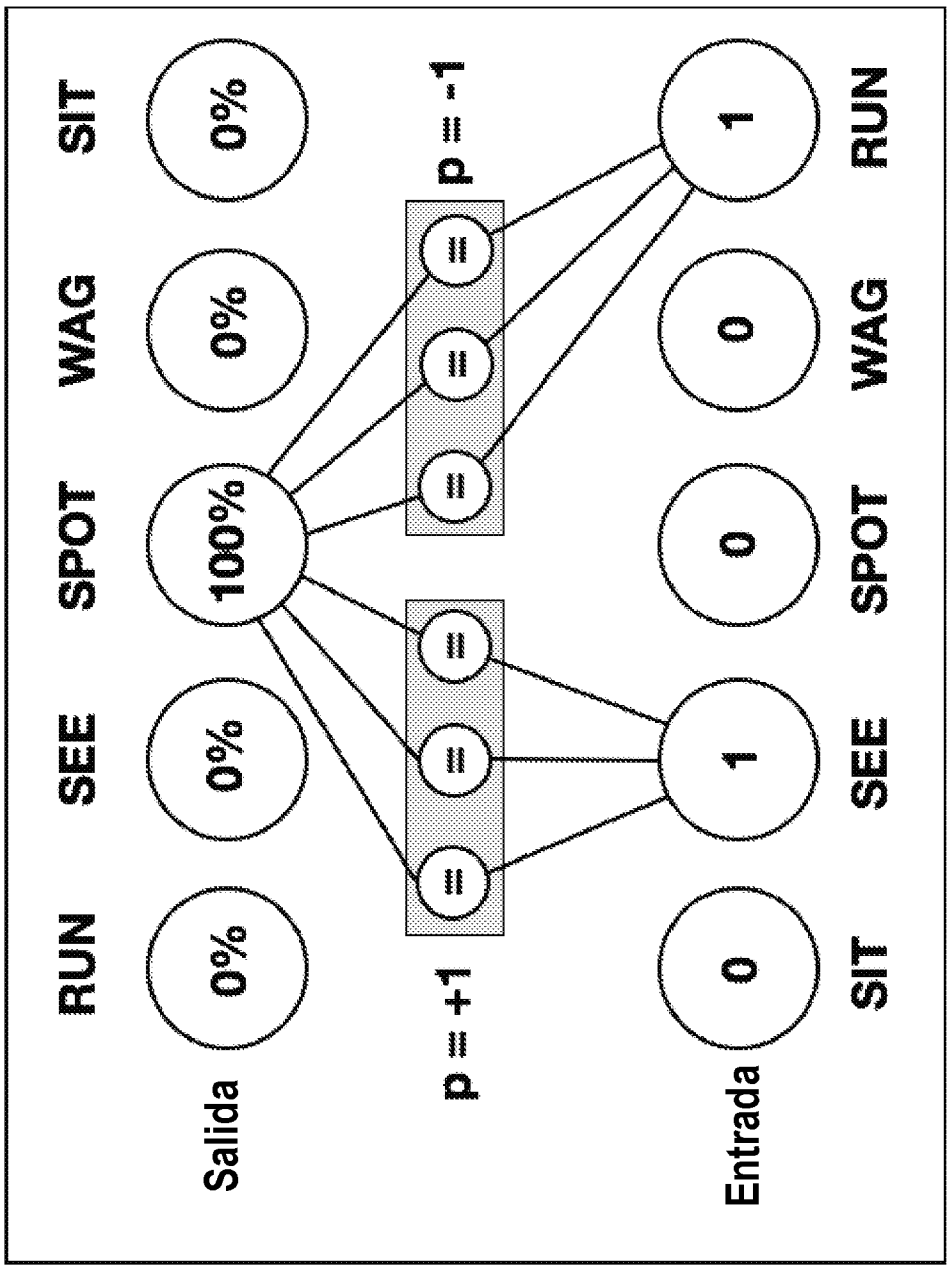
RUN
SEE
SPOT
WAG
SIT
Salida
0%
0%
100%
0%
0%
p = +1
p = -1
Entrada
0
1
0
0
1
SIT
SEE
SPOT
WAG
RUN

FIG. 4

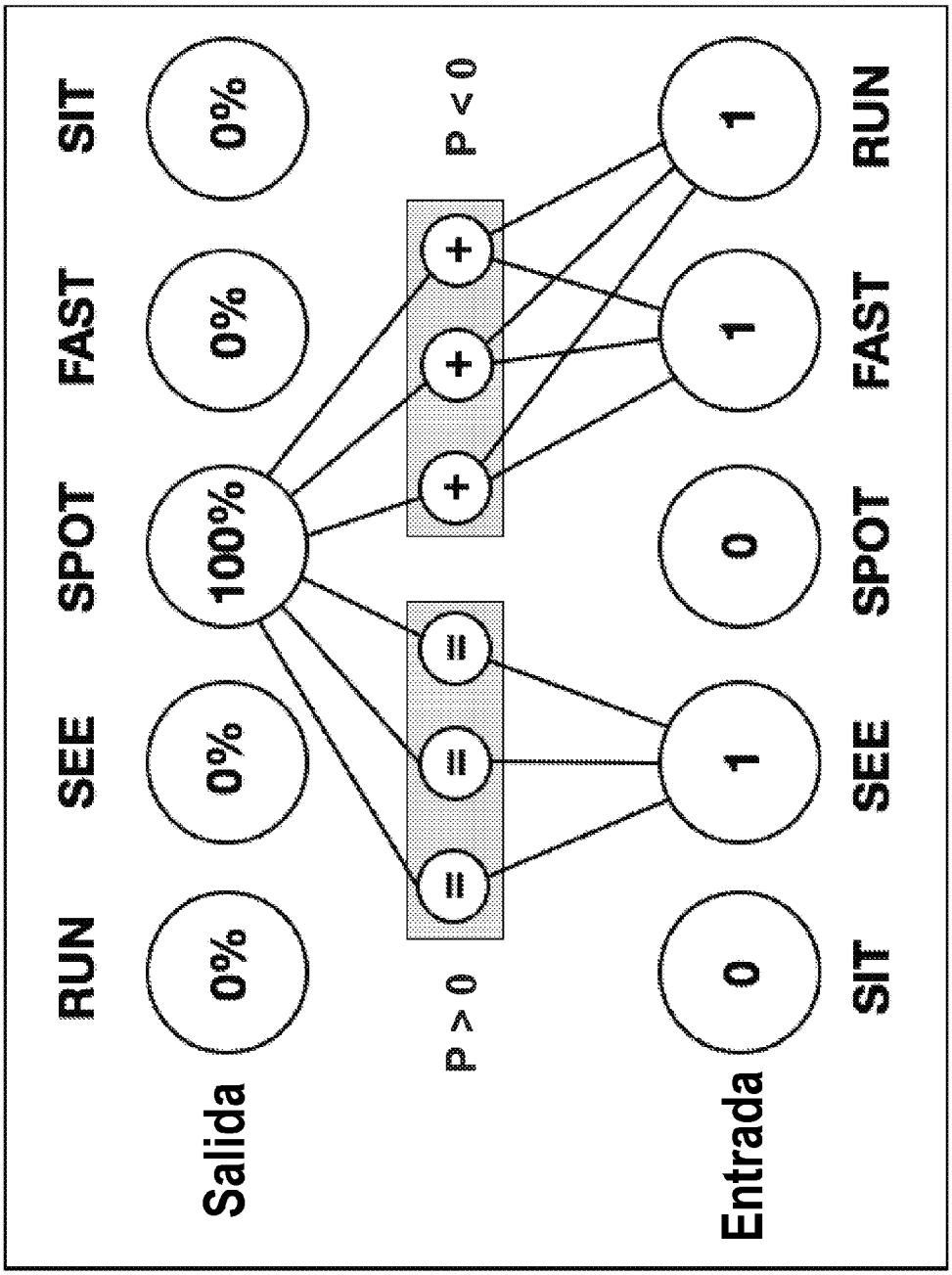
RUN
SEE
SPOT
FAST
SIT
Salida
0%
0%
100%
0%
0%
P > 0
P < 0
=
=
=
+
+
+
Entrada
0
1
0
1
1
SIT
SEE
SPOT
FAST
RUN

FIG. 5

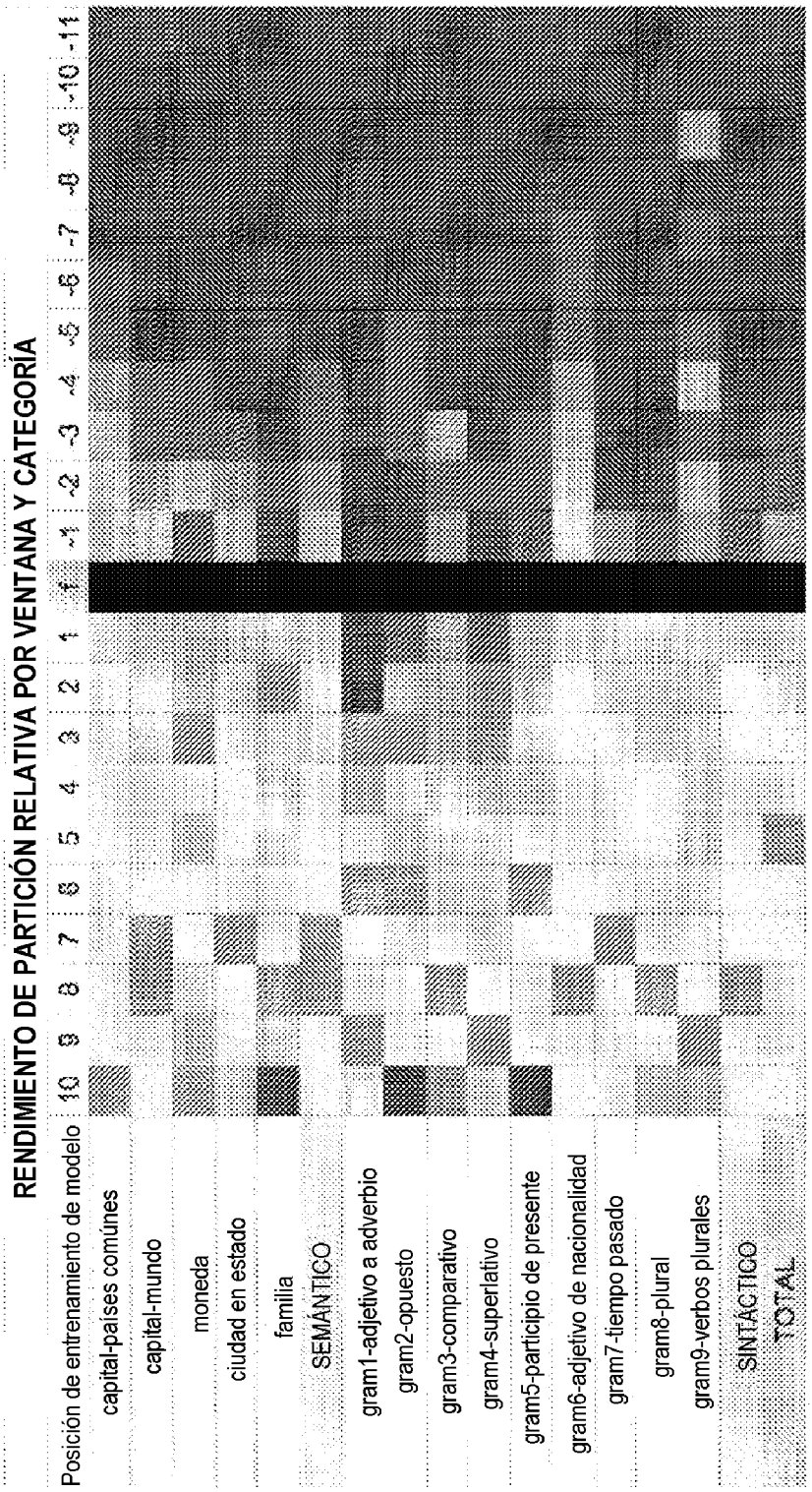
RENDIMIENTO DE PARTICIÓN RELATIVA POR VENTANA Y CATEGORÍA
Posición de entrenamiento de modelo
10 9 8 7 6 5 4 3 2 1 f -1 -2 -3 -4 -5 -6 -7 -8 -9 -10 -11
capital-países comúnes
capital-mundo
moneda
ciudad en estado
familia
SEMÁNTICO
gram1-adjetivo a adverbio
gram2-opuesto
gram3-comparativo
gram4-superlativo
gram5-participio de presente
gram6-adjetivo de nacionalidad
gram7-tiempo pasado
gram8-plural
gram9-verbos plurales
SINTÁCTICO
TOTAL

FIG. 6

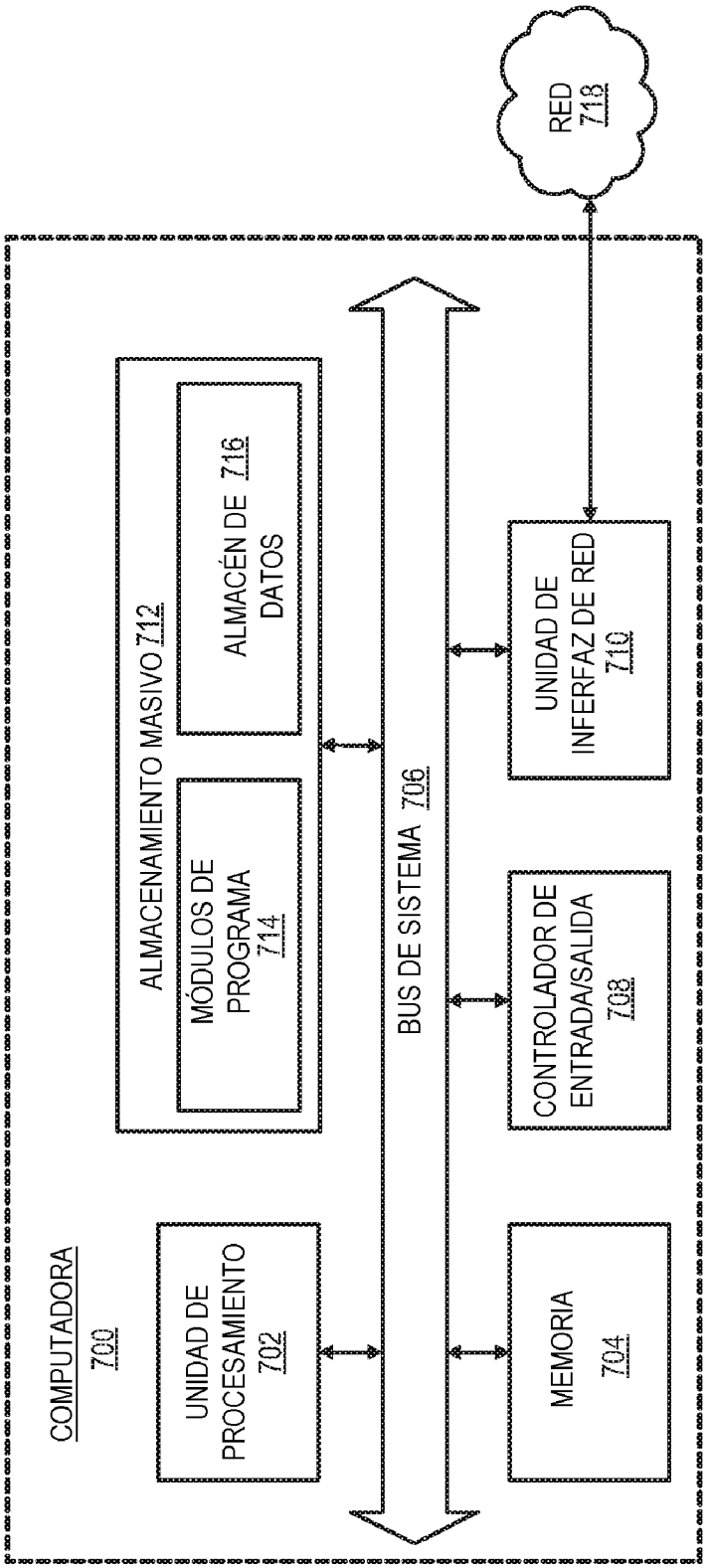
COMPUTADORA
700
UNIDAD DE PROCESAMIENTO 702
MEMORIA 704
BUS DE SISTEMA 706
CONTROLADOR DE ENTRADA/SALIDA 708
UNIDAD DE INTERFAZ DE RED 710
ALMACENAMIENTO MASIVO 712
MÓDULOS DE PROGRAMA 714
ALMACÉN DE DATOS 716
RED 718

FIG. 7