ISSN 1870-4069

Análisis y comparación de diferentes métodos de reconstrucción de árboles semánticos

Ramón Rivera Camacho, Ricardo Barrón Fernández, Hiram Calvo

Instituto Politécnico Nacional,
Centro de Investigación en Computación, Ciudad de México,
México

ulfrheimr@gmail.com,rbarron@cic.ipn.mx,hcalvo@cic.ipn.mx

Resumen. Los procesos modernos enfocados al procesamiento de lenguaje natural basados en aprendizaje profundo se apoyan en varias etapas de pre-entrenamiento. Sin embargo, hay ocasiones en las que la complejidad de las tareas requieren, después de los escenarios antes mencionados, un procesamiento adicional. En la literatura se menciona la estructuración por medio de árboles semánticos como uno de los métodos adicionales más prolíficos. En el presente trabajo comparamos los métodos de construcción de árboles semánticos por el método de duplas ordenadas y, basados en el analizador sintáctico de Stanford. Analizamos los procesos de reconstrucción de tuplas individuales y de estructuras completas para comparar la eficiencia de los métodos a través de la información semántica que cada uno puede mantener.

Palabras clave: Procesamiento de lenguaje natural, construcción de árboles semánticos, autoenconder, Stanford parser.

Analysis and Comparison of Different Methods of Reconstruction of Semantic Trees

Abstract. The addressing of natural language processing tasks by modern techniques, inspired in deep learning, rely on different pre-training stages. Nevertheless, some problems need an additional processing, after the referred scenarios, because of the complexity they exhibit. Structuring of semantic trees is one of the most prolific methods in the literature for this purpose. In this paper we compare the methods of construction of semantic trees by the method of ordered tuples, and based on the Stanford parser. We analyze the reconstruction of individual tuples and the construction of complete structures in order to compare the efficiency of both methods. We focus the measuring process in the amount of semantic information both methods can hold.

Keywords: Natural language processing, semantic tree construction, autoenconder, Stanford parser.

1. Introducción

Los métodos de aprendizaje modernos, entre los que se incluyen los basados en aprendizaje profundo, intentan omitir el tratamiento manual de los rasgos iniciales de algún problema realizando varias fases de pre-entrenamiento no supervisado. Se ha probado que estas fases proveen buenas distribuciones iniciales de los datos capturando las dependencias entre los parámetros de los mismos [3]. El uso de estos métodos dentro del ámbito del procesamiento de lenguaje natural (NLP, por sus siglas del inglés *Natural Language Processing*) no es la excepción.

Una de las técnicas más utilizadas en la fase de pre-entrenamiento es la vectorización de palabras, en la cuál cada palabra dentro del *corpus* es representada como un vector de n dimensiones. Un primer enfoque, basado en lexicones, consiste en tomar una palabra y obtenerla por medio de su respectivo identificador simbólico. Este modelo presenta varias limitaciones, la principal de ellas es la dispersión ya que la representación tendría el número de dimensiones del tamaño del vocabulario y solamente una casilla con información significativa; fenómeno conocido como *one-hot representation*.

Más aún, aquellas palabras que raramente aparezcan en el *corpus* serán escasamente estimadas y el modelo no será capaz de lidiar con palabras que no estuvieron presentes en la etapa de entrenamiento [12].

Es deseable que las palabras nos demuestren la relación con su contexto, los modelos de incrustación de palabras (*Word embeddings models*, en inglés) proveen de dicha información. Después del proceso de incrustación, una palabra a la par de su contexto puede ser representada por un vector denso de dimensión n. Las implementaciones más conocidas de éstos modelos son las basadas en redes neuronales profundas [2] o modelos *skip-gram* [7].

Inclusive aún con la presencia de varios escenarios de pre-entrenamiento, algunos problemas requieren del desarrollo de metodologías de estructuración de la información para tener un comportamiento predecible de los datos sobre la información ya distribuída coherentemente. Éstas metodologías han obtenido resultados favorables con técnicas de construcción de árboles semánticos, los cuales mantienen la máxima semántica posible a partir de reconstrucción de tuplas ordenadas generadas por modelos de incrustación de palabras [4, 10].

En dichos trabajos se presenta la pauta de una posible estructuración por medio analizadores sintácticos (*Parsers*, en inglés) tradicionales, si bien es correcto acentuar que estos mejoran sustancialmente las tareas tradicionales del NLP como el conteo de palabras o n-gramas [9], ninguna comparación formal entre las propiedades de los métodos anteriores ha sido provista.

El presente artículo se divide en las siguiente secciones. En la sección 2 se describen las propuestas de estructuración de árboles semánticos basadas en técnicas de aprendizaje profundo para ambos casos, duplas ordenadas y basadas en analizadores sintácticos. En la sección 3 se presentan los resultados de la evaluación de medidas que representen la cantidad de información semántica retenida. Finalmente, en la sección 4, se presentan las conclusiones de los resultados obtenidos.

2. Propuesta

El objetivo principal del presente trabajo es presentar una técnica para medir y comparar la pérdida semántica en la tarea de construir un árbol semántico por los métodos de analizadores sintácticos y de duplas ordenadas. Al generar estructuras semánticas expresamos la relación de las palabras en base a una función de energía, es posible medir la pérdida de energía de manera no supervisada construyendo estas estructuras y despues intentando obtener los datos iniciales, un proceso de reconstrucción.

Aquellas construcciones que presenten menor pérdida también serán las que más fácilmente serán reconstruídas y por ende, las que mayor coherencia o información semántica mantengan. Para la medición nos apoyamos en modelos de energía (EBMs, por sus siglás del inglés *Energy Based Models*) que capturan dependencias asociando una energía escalar a cada configuración de un conjunto de variables y, cuyo entrenamiento y aprendizaje puede ser utilizado como una alternativa a modelos probabilísticos para labores de predicción, clasificación y toma de decisiones [5].

En específico utilizamos *autoencoders* (AE), ya que son entrenados con mayor facilidad y rapidez en comparación a sus contrapartes basados en métodos probabilísticos [11]. Además, son ampliamente utilizados como bloques de construcción para el entrenamiento de redes profundas [1].

Este proceso se realiza en dos etapas [6, 13]: la codificación, en la cual se mapea de manera determinista el vector de entrada a una representación intermedia por medio de una función no lineal y; el proceso inverso, en el cuál se mapea de regreso la representación intermedia a un vector reconstruido, en forma similar por una función no lineal.

Debido a que la pérdida de información semántica está regida por la reconstrucción del árbol, se comparará el desempeño en este escenario para ambos métodos.

En primera instancia, se construyeron los árboles semánticos por medio de tuplas ordenadas con el proceso descrito por Huang [4] y Socher [10], de longitud dos. Las frases son separadas en palabras y después unidas en pares. El resultado es un proceso de presentación-codificación-reconstrucción con vectores de incrustación de dimensiones $[1 \times n] \rightarrow [1 \times 2n] \rightarrow [1 \times n]$, respectivamente. Repetimos el proceso hasta obtener una sola dupla que representa el árbol semántico completo. Este proceso se muestra en la Figura 1-a.

Para el segundo acercamiento utilizamos el Stanford parser[1] para obtener árboles de dependencias. Cabe mencionar que este analizador no produce estríctamente árboles binarios, por lo tanto para la construcción de duplas se utiliza un algoritmo basado de DFS como se muestra en la Figura 1-b. De igual manera, el proceso de codificación-reconstrucción mantiene el formato de dimensionalidad $[1 \times n] \rightarrow [1 \times 2n] \rightarrow [1 \times n]$, para labores de comparación.

En la Figura 2 se contrastan los métodos de construcción de estructuras de máxima semántica. Mientras que en 2-a la construcción de árboles esta

[1] http://nlp.stanford.edu/software/lex-parser.shtml

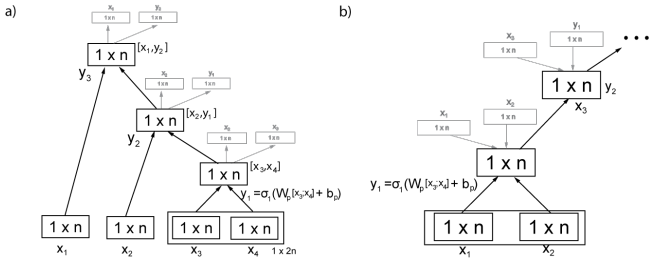

Fig. 1. Construcción de árboles semánticos por métodos para **a)** Duplas ordenadas de longitud 2 y **b)** Analizador sintactico de Stanford basado en algoritmo DFS.

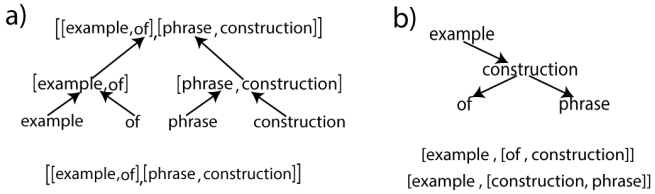

Fig. 2. Ejemplo de construcción semántica para una frase de prueba para los métodos **a)** Duplas ordenadas y **b)** Analizador sintactico.

basada en el ordenamiento semántico de las palabras, dando como resultado una dupla representativa. En 2-b se siguen las reglas pre-establecidas del analizador sintáctico, generando más de una dupla de expresión. A medida que se presenten nuevas frases los modelos de duplas resultantes tendrán mejor desempeño para aquellas construcciones que sean observadas con mayor frecuencia.

3. Experimentos y resultados

Para la construcción de vectores de incrustaciones de palabras optamos por el método basado en el algoritmo *CBOW skip-gram* propuesto por Mikolov, ya que genera mejores resultados en cuanto a exactitud y velocidad de entrenamiento que otros métodos de incrustaciones para labores como POS, NER, SRL, etc. [8].

Como apoyo para este modelo utilizaremos el conjunto de datos pre-entrenado de noticias provisto por Google en su implementación *word2vec*[2], el cuál contiene 100 mil millones de palabras y su vocabulario es de 3 millones de palabras.

Para el entrenamiento del AE se utilizará el error cuadrático medio (MSE, por sus siglas del inglés *Mean Squared Error*) como función de costo J, para

[2] https://code.google.com/archive/p/word2vec/

cada nodo del árbol resultante T. El error total será la suma de todos los errores de cada nodo de la estuctura, como se muestra en (1):

$$J^i(W,b) = \sum_{d \in T^i} \frac{1}{2} ||x_{l,r} - x'_{l,r},||^2, \tag{1}$$

donde

d cada dupla de la representación del árbol correspondiente a la frase **i**,
$x_{l,r}$ dupla de un nodo dado,
$x'_{l,r}$ representación de la reconstrucción de un nodo dado.

La tarea principal en la etapa de entrenamiento es minimizar la función de costo J para el conjunto de frases I dada una razón de aprendizaje λ, como se muestra en (2):

$$\min_{W,b} \sum_{i \in I} J^i(W,b) + \frac{\lambda}{2} ||W||^2, \tag{2}$$

donde

$i \in I$ cada una de las frases del conjunto de datos **I**,
W, b matriz de pesos sinápticos y de compensadores de la red neuronal, respectivamente.

Para los experimentos utilizamos textos obtenidos del sitio de internet Experience Project[3]. El conjunto tiene un total de ∼3.3K frases. Utilizamos este *corpus* ya que presenta errores de sintaxis y gramaticales, además de que contiene etiquetas no textuales. El conjunto contiene 5806 palabras distintas con una relación de palabras desconocidas con respecto al modelo de referencia pre-entrenado de ∼9 %, por lo tanto se espera un rendimiento menor al que se obtendría si estuviera propiamente escrito.

El pre-procesamiento de los datos es de la manera más simple posible, solamente se retiran todos los elementos no textuales y se separan frases de párrafos extensos, ninguna labor extra es realizada.

Al término de la separación de conjuntos en razón 5:1:1, tenemos un total de $\sim 2.4K$ frases para entrenamiento y de ~ 500 frases para las etapas de validación y pruebas.

La comparación está basada en dos principales medidas. La reconstrucción promedio del MSE de duplas y la reconstrucción total promedio de árboles. Mientras que la primera de ellas provee la capacidad del método para obtener valores significativos de reconstrucción a la hora de entrenar, la segunda indica la eficiencia del método para evaluar la estructura completa en etapa de pruebas.

En la Figura 3 se muestran los resultados de la reconstrucción de duplas. En este escenario el método de duplas ordenadas tiene mejor rendimiento que su contraparte. Mientras que éste reporta un mínimo de ∼0.069, el basado en analizadores sintácticos reporta un mínimo de ∼0.081, ambos restringidos hasta 50 épocas. Con lo que también se infiere una convergencia más rápida.

[3] http://www.experienceproject.com/

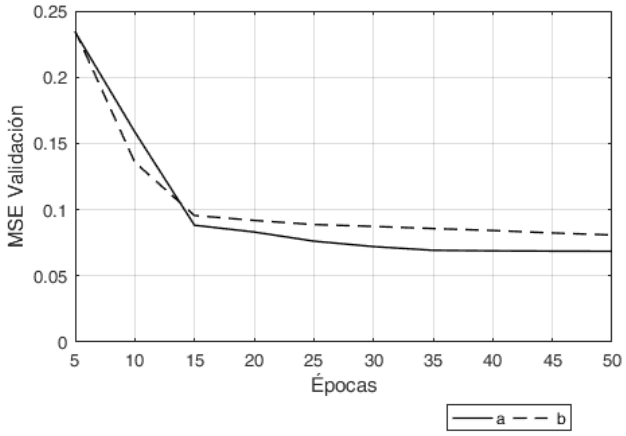

Fig. 3. MSE promedio de reconstrucción de duplas para los métodos **a)** duplas ordenadas y **b)** basado en analizadores sintácticos.

Para el escenario de reconstrucción de árboles se realiza un conjunto de 30 experimentos sobre *batches* de palabras. Con el fin de tener un rango más amplio de conjuntos de pruebas, se varían las condiciones de parada temprana y cantidad de nodos en rangos de 20 a 50 nodos. Se reportan los errores de reconstrucción total promedio para cada experimento en la Figura 4.

Después de obtener la media de todos los experimentos expuestos previamente, se aprecia que el método basado en duplas ordenadas ofrece mejores resultados para la reconstrucción total de árboles con una media de $\sim$0.127 contra la media de $\sim$0.171 para el acercamiento por medio de analizadores sintácticos.

Un mejor rendimiento es obtenido en ambas etapas de comparación para el método de duplas ordenadas. Para la etapa de reconstrucción de duplas se reporta un $\sim$ 17 % de mejoría con respecto al acercamiento de analizadores sintácticos, mientras que para labores de reconstrucción total de árboles la reconstrucción de duplas mejora al reportar $\sim$ 35 % contra el acercamiento de analizadores sintácticos.

4. Conclusiones

En este trabajo de investigación se compararon dos métodos de construcción de árboles, el primero basado en duplas ordenadas y el segundo, por medio del analizador sintáctico de Stanford para los escenarios de reconstrucción de duplas individuales y de error de reconstrucción de todo el conjunto de árboles en el *corpus* de prueba.

Se encontró que para ambos casos el desempeño del primer método es mejor. Estos resultados son justificables debido a la dispersión de las tuplas. Se debe recordar que el objetivo principal de un analizador sintáctico es mostrar la

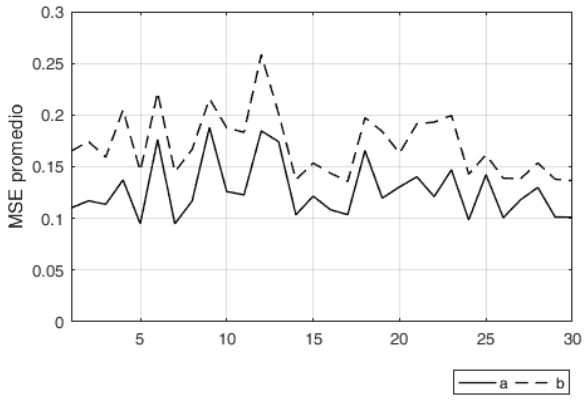

Fig. 4. MSE de reconstrucción total para rangos de parada y nodos de AE para métodos **a)** duplas ordenadas y **b)** basado en analizadores sintácticos.

estructura gramatical de una frase en específico. Al tener un conjunto de datos de prueba carente de éstas características la labor se convierte una tarea difícil para un analizador y por lo tanto, los árboles resultantes no siempre serán reconstruidos de la misma manera.

Por otra parte, ya que el método de estructuración por medio de duplas no lidia con clases gramaticales ni parecidos, a medida que los errores aparezcan serán obviados, lo cual se soporta con el conjunto restante de árboles bien estructurados.

Finalmente, se destaca que mientras el método de duplas ordenadas trabaja calculando el error mínimo por nivel para la construcción de los árboles, el método de analizador de Stanford utiliza la estructura procesada y calcula el error de reconstrucción dado el árbol completo.

Agradecimientos. Este trabajo fue realizado gracias al apoyo del Instituto Politécnico Nacional, los proyectos SIP-IPN 20162058 y 20160828 y COFAA-IPN.

Referencias

1. Bengio, Y.: Learning Deep Architectures for AI. Foundations and Trends in Machine Learning 2(1) (2009)
2. Collobert, R., Weston, J.: A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In: Proceedings of the 25 th International Conference on Machine Learning (2008)
3. Erhan, D., Bengio, Y., Courville, A., Vincent, P., Bengio, S.: Why Does Unsupervised Pre-training Help Deep Learning? Journal of Machine Learning Research 11, 625–660 (2010)

4. Huang, E.: Paraphrase Detection Using Recursive Autoencoder (2011)
5. Lecun, Y., Chopra, S., Hadsell, R., Ranzato, M.A., Huang, F.J.: A Tutorial on Energy-Based Learning. Predicting Structured Data, MIT Press (2006), `http://yann.lecun.com`
6. LISA Lab: Deep Learning Tutorial (2015)
7. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient Estimation of Word Representations in Vector Space (2013)
8. Schnabel, T., Labutov, I., Mimno, D., Joachims, T.: Evaluation methods for unsupervised word embeddings. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp. 298–307. Association for Computational Linguistics (2015)
9. Sidorov, G., Velasquez, F., Stamatatos, E., Gelbukh, A., Chanona-Hernández, L.: Syntactic N-grams as machine learning features for natural language processing. Expert Systems with Applications: An International Journal 41(3), 853–860 (2014)
10. Socher, R., Pennington, J., Huang, E.H., Ng, A.Y., Manning, C.D.: Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions. In: EMNLP '11 Proceedings of the Conference on Empirical Methods in Natural Language Processing. pp. 151–161 (2011)
11. Swersky, K., Ranzato, M.A., Buchman, D., Marlin, B.M., De Freitas, N.: On Autoencoders and Score Matching for Energy Based Models. In: Proceedings of the 28 th International Conference on Machine Learning (2011)
12. Turian, J., Ratinov, L., Bengio, Y.: Word representations: A simple and general method for semi-supervised learning. In: 48th Annual Meeting of the Association for Computational Linguistics (2010)
13. Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.A.: Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. Journal of Machine Learning Research 11 (2010)