Plataforma de exploración de la Composición Semántica a partir de Modelos de Lenguaje pre-entrenados y embeddings estáticos

Platform for exploring Semantic Composition from pre-trained Language Models and static embeddings

Adrián Ghajari[1], Víctor Fresno[1] and Enrique Amigó[1]

[1] *Universidad Nacional de Educación a Distancia (UNED), España*

Abstract

El crecimiento de la capacidad de procesamiento y el advenimiento del modelo Transformer han modificado el panorama del PLN. El proceso conocido como Transferencia de Aprendizaje ha facilitado la consecución de resultados cercanos al estado-del-arte a una fracción del coste computacional. En este ámbito, este artículo presenta una aplicación cliente-servidor capaz de obtener vectores contextualizados (o estáticos) de palabras dentro de textos y a partir de una gran cantidad de modelos pre-entrenados, realizar composición semántica para, finalmente, visualizar en un espacio tridimensional las representaciones obtenidas y estimar su similitud semántica; todo esto, explotando los recursos hardware disponibles.

English translation. The computing power growth and the advent of the Transformer model have changed the NLP landscape. Transfer Learning has allowed the posibility of achieving state-of-the-art results at a fraction of the computational cost. In this scope, this work presents the development of a server-client application capable of obtaining contextual and static word vectors from a wide variety of models, operate with them to achieve semantic composition to, lastly, visualize them in a 3-dimensional space and obtain semantic similarity; all of this, while exploiting the hardware resources available.

Keywords

Composición semántica, Vectores de frases, Transformers,

1. Introducción

La llegada del modelo Transformer [1] ha revolucionado el área del NLP, fundamentalmente por su capacidad de aplicar patrones aprendidos durante su entrenamiento sobre distintas tareas nuevas, aunque relacionadas, lo que se conoce como Transferencia de Aprendizaje (TA). Este modelo captura información sobre el contexto en el que se encuentra cada palabra dentro de una frase, generando representaciones vectoriales de las mismas como paso intermedio antes de un proceso de ajuste fino (fine tuning). Estas representaciones de palabras se pueden procesar para realizar Composición Semántica. El Principio de Composicionalidad está basado en que el significado del todo es una función del significado de sus partes y de cómo están sintácticamente combinadas; por su parte, el Principio de Contextualidad afirma que el significado de las unidades lingüísticas emerge del contexto en el que se usan. Se conoce como Composición semántica al proceso por el que se generan representaciones vectoriales de frases a partir de los significados individuales de sus palabras constituyentes y de cómo estas se combinan.

En este artículo se presenta una plataforma software[1] que realiza composición semántica a partir de modelos pre-entrenados de los repositorios de HuggingFace[2] (contextuales) y Gensim[3] (estáticos), utilizando textos facilitados por el usuario, siendo capaz de representar los resultados en el espacio tridimensional, permitiendo así la comparación de embeddings en diferentes tareas de similitud semántica (STS), o estudiar el efecto de la contextualidad en este tipo de problemas, al permitir trabajar con representación a diferentes capas internas de la red.

2. Motivación

Se han desarrollado numerosos marcos de evaluación, nacidos de la necesidad de cuantificar el éxito de los modelos, sin embargo, todos estos marcos están orientados a asignar un valor de rendimiento o precisión en un marco de referencia arbitrario. El

SEPLN-PD 2022. Annual Conference of the Spanish Association for Natural Language Processing 2022: Projects and Demonstrations, September 21-23, 2022, A Coruña, Spain

aghajari@lsi.uned.es (A. Ghajari); vfresno@lsi.uned.es (V. Fresno); enrique@lsi.uned.es (E. Amigó)

[1] https://github.com/adriangh-ai/AllSpark
[2] https://huggingface.co
[3] https://radimrehurek.com/gensim/

problema que presentan es que esta información no nos permite comprender el funcionamiento del modelo a evaluar o cómo consigue capturar el lenguaje para el que se ha entrenado.

El éxito de la TA ha traído consigo una sobrecarga de modelos; sólo en el repositorio de HuggingFace hay más de 34 mil almacenados. Asimismo, la composición semántica sobre la salida de los modelos de lenguaje se enfrenta a problemas tales como la degradación de representación [2, 3, 4], o si realmente capturan la información semántica del texto de entrada. Esto hace abstrusa la evaluación y establecimiento de métricas, más allá de la inspección supervisada del resultado. Los objetivos de este trabajo son el análisis e implementación de mecanismos de composición semántica, con una interfaz que sirva como capa de abstracción para la búsqueda, obtención y almacenamiento de diversos modelos de representación basados en RNA como un marco de trabajo para el control y visualización de los diferentes modelos. Lo anterior provee de un mecanismo para el estudio de la estructura interna de modelos de lenguaje, al permitir observar representaciones provenientes de la salida de capas intermedias con las herramientas de reducción dimensional implementadas para la visualización 3D de embeddings n-dimensionales; así como el estudio comparativo de modelos pre-entrenados y métodos de composición mediante métricas de similitud semántica.

3. Funcionalidad y Caso de Uso

Su función principal es la obtención de representaciones vectoriales a partir del modelo de lenguaje neuronal descargado desde un repositorio. La selección de capas internas del modelo a partir de las cuales obtener la composición semántica mediante distintos mecanismos (suma, media aritmética, [CLS] token y F_{inf}, F_{joint}, F_{ind} de ICDS [5]) y la posibilidad de procesar de forma concurrente y con paralelismo de datos el conjunto de muestra. Finalmente, la visualización de las frases en una gráfica 3D interactiva. Se trata de una aplicación que puede ejecutarse con independencia de despliegue del cliente y servidor, pudiendo encontrarse y explotar recursos en máquinas locales o estaciones de trabajo remotas y distintos sistemas operativos.

El usuario tendrá conocimiento esencial sobre modelos de lenguaje y composición y se intentan cubrir los siguientes casos de usos diferenciados:

Inferencia sobre una muestra de datos dada. Obtención de la composición de uno o más conjuntos de frases, bien para su visualización o para su uso en otra tarea.

Recuperación de sesión anterior. Volver a cargar una o varias sesiones anteriores para su visualización y comparación.

4. Arquitectura general

Modelo cliente-servidor multi-plataforma, con *back-end* escrito en Python y *front-end* en ElectronJS[4] y Plotly DASH[5] con comunicación remota basada en gRPC protobuf[6].

4.1. Servidor

El servidor contiene la lógica relacionada con la gestión de modelos y el procesamiento de la evaluación, así como la composición semántica, pudiendo ejecutarse en una máquina remota. Es quien implementa la definición de la interfaz gRPC para ofrecer servicios a clientes, gestiona el almacenamiento de modelos y mantiene la relación hardware del sistema. Por último, procesa la entrada de texto, inferencia y composición semántica.

Módulo de sesión Realiza las tareas de inferencia y composición individuales mediante multiprocesamiento, haciendo uso de los módulos de modelos y composición. Se ha implementado paralelismo de datos instanciando el modelo en cada dispositivo con hilo exclusivo y dividiendo la carga a partes iguales entre dispositivos, mostrando mejor rendimiento frente a Pytorch DataParallel. A su vez, la técnica de Uniform Length Batching evita el procesamiento de tokens [PAD] innecesarios mediante ordenación y agrupación en batches de frases según longitud.

Módulo de modelos y composición Instanciarán un objeto con los métodos necesarios para la inferencia y la composición que se asignarán a *workers*; se ha optado por la eliminación de los tokens que no se corresponden con una palabra o un fragmento de palabra con una función de limpieza que convierte en una máscara los identificadores de los tokens especiales. Asimismo, en caso de haber seleccionado un rango de capas para su procesado, se operará la media aritmética sobre los resultados de composición individuales por capa.

[4] https://www.electronjs.org/
[5] https://plotly.com/
[6] https://grpc.io/

Figure 1: Pantalla principal.

4.2. Cliente

Contiene la interfaz de usuario y los métodos gRPC de comunicación con el servidor. Se ocupa del preprocesamiento de los datos, así como la lógica que atiende a la reducción dimensional y representación de resultados. Se divide en dos bloques funcionales gobernados por ElectronJS y DASH, que se comunican entre ellos por HTTP mediante un Web Server Gateway Interface, Waitress[7], para proveer la interfaz. Tras el lanzamiento y la conexión a un servidor externo o ejecución y conexión local, se llega a la pantalla de ajuste y selección de parámetros de inferencia. En su inicio la aplicación cliente actualiza la relación de dispositivos y modelos disponibles del servidor, o para su descarga desde los almacenes de HuggingFace y Gensim.

Pestaña Principal Los modelos disponibles (contextuales y estáticos) se ofrecerán en forma de lista predictiva al introducir texto en el área de búsqueda. Una vez descargados podrá elegirse la salida de una de las capas del mismo, o un rango de ellas. Asimismo, se ofrece para su selección una relación de métodos de composición semántica y la lista de dispositivos de computación encontrados en el servidor para su asignación.

El área de selección de archivo acepta diversos formatos: estructurados, como csv, json o excel, y texto desestructurado en txt. En este último caso, el sistema tratará de reconocer las frases que contiene el texto a través de la librería Natural Language Toolkit (NLTK)[8]. Finalmente, se podrán seleccionar columnas, que serán visualizadas en la misma gráfica con colores distintos por columna. Por otro lado, los archivos de sesiones anteriores guardados se pueden volver a cargar y visualizar.

Tras seleccionar la configuración completa, pueden añadirse a la lista de peticiones de inferencia. Es posible añadir y borrar cuantas peticiones se desee; pulsando el botón de lanzamiento de inferencia, serán procesadas en el servidor de forma concurrente en los dispositivos que cada uno tenga asignados.

Pestañas de inferencia: Tras el proceso de evaluación, el servidor envía los datos al cliente, que los mostrará en una nueva pestaña de inferencia. En

[7] https://github.com/Pylons/waitress

[8] https://www.nltk.org/

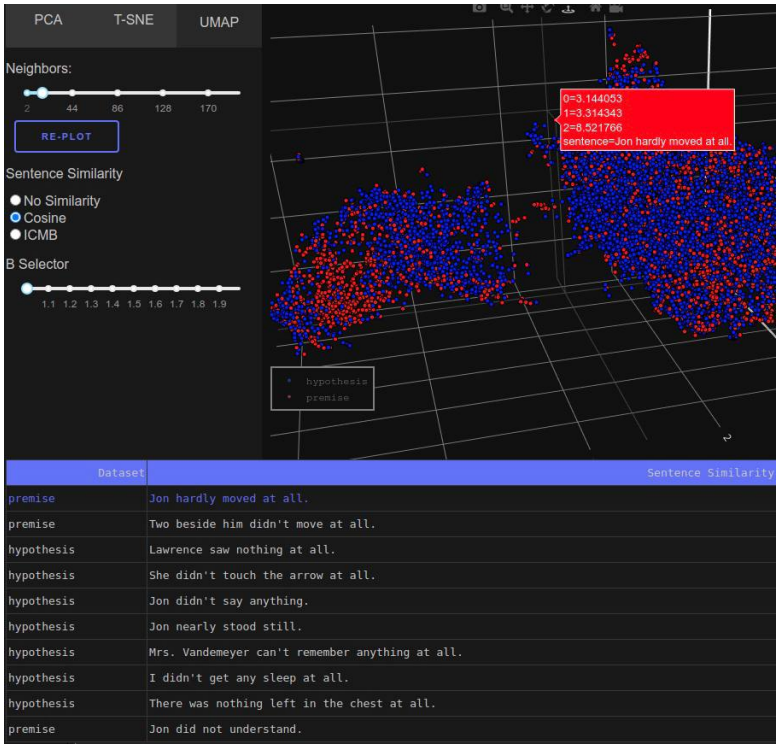

Figure 2: Pestaña de inferencia.

ella se ofrecen distintos métodos de reducción dimensional, t-SNE [6], Principal Component Analysis [7] y UMAP [8], seleccionables como subpestañas, que contienen los elementos de ajuste de parámetros de cada uno. Desde la misma es también posible guardar los vectores resultado del proceso de composición El resultado de estos métodos se mostrará en la gráfica, que ofrecerá una representación interactiva y tridimensional por color según columna de procedencia en la muestra original con los datos de cada frase. Seleccionando el método de similitud semántica, como similitud coseno, y un punto en la gráfica de representación, se mostrará una tabla con las 10 frases más cercanas en el conjunto de origen, así como la columna a la que pertenecen.

5. Ejemplo de uso

Es en esta pantalla (ver Figura1) el usuario puede seleccionar los dispositivos de computación a usar, descargar y seleccionar modelos y capas a procesar, ver la estimación de ocupación de memoria, elegir método de composición y cargar el archivo de muestras. Tras la selección, se procede a la inferencia, añadiendo los resultados a una nueva pestaña.

Finalmente (ver Figura2), pueden elegirse distintos métodos de reducción dimensional (esquina superior izquierda), modificar sus parámetros de operación (izquierda, ver Figura3), visualizar las frases más cercanas a un punto (tabla de similitud coseno, parte inferior de la imagen con la frase seleccionada marcada por una etiqueta) y guardar los resultados de la sesión. A modo de ejemplo, se ofrecen los puntos correspondientes a las columnas *hipótesis* (azul) y *premisa* (rojo) del conjunto de datos GLUE, subset *mnli*, según la última capa del modelo BERT; puede observarse empíricamente la posición relativa entre frases, distancia y agrupación, según la función de composición semántica, el modelo y capa del mismo elegidos.

6. Conclusiones y trabajos futuros

Este trabajo presenta una aplicación distribuida multiplataforma cuya finalidad es la asistencia a la investigación en el estudio de la composición a partir de modelos de lenguaje neuronales. Se han implementado algoritmos de composición semántica, reducción de dimensionalidad y similitud semántica, además de técnicas de optimización de inferencia.

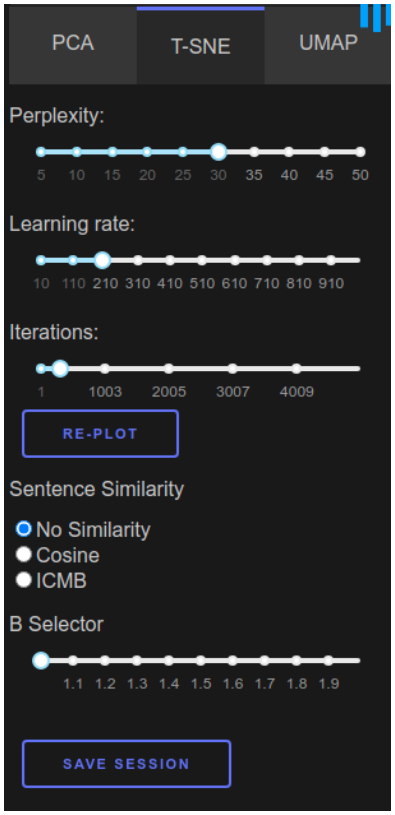

Figure 3: Detalle de modificación de parámentros t-SNE.

En lo relativo a futuras funcionalidades, se abordará el paralelismo de un solo modelo en inferencia, dividiendo el modelo en capas y repartiéndolas entre dispositivos. Adicionalmente, se pretenden implementar soluciones propuestas al problema de la isometría semántica de las representaciones, evidenciado en los trabajos [2, 9]. Finalmente, algunos estudios apuntan a la posibilidad de que distintas cabezas de auto-atención del modelo Transformer atienda a distintos aspectos semánticos, [10]; aislarlos y generar su representación podría ofrecer una nueva perspectiva.

Acknowledgments

Este trabajo ha sido financiado por los proyectos del Ministerio de Ciencia y Innovación DOTT-HEALTH (PID2019-106942RB-C32), gracias al acuerdo UNED - Ministerio de Economía y Competitividad de España con ref. C039/21-OT, y MISMIS project (PGC2018-096212-B), así como por el proyecto PID2020 GID2016-39 de Innovación Docente de la Universidad Nacional de Educación a Distancia y los proyectos del Consejo Europeo de Investigación (ERC) bajo el programa de investigación e innovación H2020-INFRAIA-2020-1: LyrAIcs (con Grant agreement Nº [964009]) y CLS-INFRA: Computational Literary Studies Infrastructure (con Grant agreement Nº [101004984]).

References

[1] A. Vaswani, G. Brain, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention Is All You Need, Technical Report, 2017.

[2] K. Ethayarajh, How contextual are contextualized word representations? Comparing the geometry of BERT, ELMO, and GPT-2 embeddings, EMNLP-IJCNLP 2019 (2020) 55–65. doi:10.18653/v1/d19-1006. arXiv:1909.00512.

[3] B. Li, H. Zhou, J. He, M. Wang, Y. Yang, L. Li, On the sentence embeddings from pre-trained language models, arXiv (2020). doi:10.18653/v1/2020.emnlp-main.733. arXiv:2011.05864.

[4] J. Gao, D. He, X. Tan, T. Qin, L. Wang, Y. Liu, Representation Degeneration Problem in Training Natural Language Generation Models, Technical Report, 2019. arXiv:1907.12009v1.

[5] E. Amigó, A. Ariza, V. Fresno, M. A. Martí, Information-teoretic compositional distributional semantics (IN PRESS) (2021).

[6] L. Van Der Maaten, G. Hinton, Visualizing Data using t-SNE, Journal of Machine Learning Research 9 (2008) 2579–2605.

[7] H. Hotelling, Analysis of a complex of statistical variables into principal components., J. of Educational Psychology 24 (1933) 498–520.

[8] L. Mcinnes, J. Healy, J. Melville, UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction (2020). arXiv:1802.03426v3.

[9] E. Amigó, F. Giner, J. Gonzalo, M. Verdejo, On the foundations of similarity in information access., Information Retrieval 23, Issue 3 (2020) 216–254.

[10] A. Rogers, O. Kovaleva, A. Rumshisky, A Primer in BERTology: What We Know About How BERT Works, Technical Report, 2020. URL: https://github.com/. arXiv:2002.12327v3.