ESTEEM: A Novel Framework for Qualitatively Evaluating and Visualizing Spatiotemporal Embeddings in Social Media

Dustin Arendt[1] **and Svitlana Volkova**[2]
[1]Visual Analytics, [2]Data Science and Analytics
National Security Directorate
Pacific Northwest National Laboratory
Richland, WA 99354
firstname.lastname@pnnl.gov

Abstract

Analyzing and visualizing large amounts of social media communications and contrasting short-term conversation changes over time and geolocations is extremely important for commercial and government applications. Earlier approaches for large-scale text stream summarization used dynamic topic models and trending words. Instead, we rely on text embeddings – low-dimensional word representations in a continuous vector space where similar words are embedded nearby each other.

This paper presents ESTEEM,[1] a novel tool for visualizing and evaluating spatiotemporal embeddings learned from streaming social media texts. Our tool allows users to monitor and analyze query words and their closest neighbors with an interactive interface. We used state-of-the-art techniques to learn embeddings and developed a visualization to represent dynamically changing relations between words in social media over time and other dimensions. This is the first interactive visualization of streaming text representations learned from social media texts that also allows users to contrast differences across multiple dimensions of the data.

1 Motivation

Social media is an example of high volume dynamic communications. Understanding and summarizing large amounts of streaming text data is extremely challenging. Traditional techniques that rely on experts, keywords and ontologies do not scale in this scenario. Dynamic topic models, trending topics are widely used as text stream summarization techniques but they are biased and do not allow exploring dynamically changing relationship between concepts in social media or contrasting them across multiple dimensions.

Text embeddings represent words as numeric vectors in a continuous space, where words within similar contexts appear close to one another (Harris, 1954). Mapping words into a lower-dimensional vector space not only solves the dimensionality problem for predictive tasks (Mikolov et al., 2013a), but also goes beyond topics and word clouds by capturing word similarities on syntactic, semantic and morphological levels (Gladkova and Drozd, 2016).

Most past work has learned text representations from static corpora and visualized[2] the relationships between embedding vectors, measured using cosine or Euclidian distance similarity, using Principal Component Analysis (PCA) projection in 2D (Hamilton et al., 2016b; Smilkov et al., 2016) or t-Distributed Stochastic Neighbor Embedding (t-SNE) technique (Van Der Maaten, 2014). Unlike static text corpora, in dynamically changing text streams the associations between words are changing over time e.g., days (Hamilton et al., 2016b,a), years (Kim et al., 2014) or centuries (Gulordava and Baroni, 2011). These changes are compelling to evaluate quantitatively, but, given the scale and complexity of the data, interesting findings are very difficult to capture without qualitative evaluation through visualization.

Moreover, the majority of NLP applications are using word embeddings as features for downstream prediction tasks e.g., part-of-speech tagging (Santos and Zadrozny, 2014), named entity recognition (Passos et al., 2014) and dependency

[1]Demo video: http://goo.gl/3N9Ozj

[2]TensorBoard Embedding Visualization: https://www.tensorflow.org/get_started/embedding_viz

parsing (Lei et al., 2014). However, in the computational social sciences domain, embeddings are used to explore and characterize specific aspects of a text corpus by measuring, tracking and visualizing relationships between words. For example, Bolukbasi et al. (2016) evaluate cultural stereotypes between occupation and gender, Stewart et al. (2017) predicted short-term changes in word meaning and usage in social media.

In this paper we present and publicly release a novel tool ESTEEM[3] for visualizing text representations learned from dynamic text streams across multiple dimensions e.g., time and space.[4] We present several practical use cases that focus on visualizing text representation changes in streaming social media data. These include visualizing word embeddings learned from tweets over time and across (A) geo-locations during crisis (Brussels Bombing Dataset), (B) verified and suspicious news posts (Suspicious News Dataset).

2 Background

2.1 Embedding Types

Most existing algorithms for learning text representations model the context of words using a continuous bag-of-words approach (Mikolov et al., 2013a), skip-grams with negative sampling (Mikolov et al., 2013b) – Word2Vec,[5] modified skip-grams with respect to the dependency tree of the sentence (Levy and Goldberg, 2014), or optimized ratio of word co-occurrence probabilities (Pennington et al., 2014) – GloVe.[6]

2.2 Embedding Evaluation

There are two principle ways one can evaluate embeddings: (a) intrinsically and (b) extrinsically.

(a) *Intrinsic evaluations* directly test syntactic or semantic relationships between the words, and rely on existing NLP resources e.g., WordNet and subjective human judgements e.g., crowdsourcing.

(b) *Extrinsic methods* evaluate word vectors by measuring their performance when used for downstream NLP tasks e.g., dependency parsing, named entity recognition (Passos et al., 2014; Godin et al., 2015).

Recent work suggests that intrinsic and extrinsic measures correlate poorly with one another (Schnabel et al., 2015; Gladkova and Drozd, 2016; Zhang et al., 2016). In many cases we want an embedding not just to capture relationships within the data, but also to do so in a way which can be usefully applied. In these cases, both intrinsic and extrinsic evaluation must be taken into account.

3 Use Cases

For demonstration purposes we rely on the Word2Vec implementation in gensim, but our tool can take any type of pre-trained embedding vectors. To ensure the quality of embeddings learned from social media streams, we lowercased, tokenized and stemmed raw posts,[7] and also applied standard NLP preprocessing to clean noisy social media texts e.g., remove punctuation, mentions, digits, emojis etc. Below we discuss two Twitter datasets we collected to demonstrate our tool for visualizing spatiotemporal text representations.

3.1 Brussels Bombing Dataset

We collected a large sample of tweets (with geo-locations and language IDs assigned to each tweet) from 240 countries in 66 languages from Twitter. Data collection lasted two weeks, beginning on March 15th, 2016 and ending March 29th, 2016. We chose this 15 day period because it includes the attacks on Brussels on March 22 (a widely-discussed event) as well as one whole week before and after the attacks. We used 140 million tweets in English to learn daily spatiotemporal embeddings over time and across 10 European countries.

Dimensions	Tweets
Belgium	1,795,906
France	7,627,599
Germany	5,186,523
Ireland	4,866,775
Spain	5,743,715
United Kingdom	81,733,747
Verified News	9,618,825
Suspicious News	8,492,905

Table 1: Brussels and news dataset statistics: the number of tweets we used to learn embeddings.

3.2 Suspicious News Dataset

We manually constructed a list of trusted news accounts that tweet in English and checked

[3]Live demo: http://esteem.labworks.org

[4]Code: https://github.com/pnnl/esteem/

[5]Word2Vec in gensim: https://radimrehurek.com/gensim/models/word2vec.html

[6]GloVe: https://cran.r-project.org/web/packages/text2vec/vignettes/glove.html

[7]Stemming is rarely done when learning embeddings. We stemmed our data because we are not interested in recovering syntactic relationships between the words.

whether they are verified on Twitter. The example verified accounts include @cnn, @bbcnews, @foxnews. We found the list of accounts that spread suspicious news – propaganda, clickbait, hoaxes and satire,[8] e.g., @TheOnion, @ActivistPost, @DRUDGE_REPORT. We collected retweets generated in 2016 by any user that mentions one of these accounts and assigned the corresponding label propagated from suspicious or trusted news sources. In total, we collected 9.6 million verified news posts and 8.4 million suspicious news tweets. We used 18 million tweets to learn monthly embeddings over time and across suspicious and verified news account types.

4 Visualization

Our objective was to provide users with a way to to visually understand how embeddings are changing across multiple dimensions. Lets consider the Brussels Twitter dataset as an example where text representations vary over time and space. We accomplish this by allowing the user to query our tool with a given keyword across set of locations, which produces corresponding visual representations of the embeddings across time and space. The user can then inspect these visual embedding representations side by side, or combine them into a single representation for a more explicit comparison across regions.

4.1 Design

The main challenge we faced in designing dynamic embedding representations was with the scale and complexity of the embeddings, which have tens of thousands of words and hundreds of dimensions. Existing embedding visualization techniques have primarily relied on scatter plot representations of projected data (Hamilton et al., 2016b), using principal components analysis or other dimension reduction techniques e.g., t-Distributed Stochastic Neighbor Embedding.

However, these techniques are problematic because they can create visual clutter if too many entities are projected, and they can be difficult to interpret. Embeddings, having high dimension, can not necessarily be projected into a 2- or 3- dimensional space without incurring significant visual distortion, which can degrade users' trust in the visualization (Chuang et al., 2012). Furthermore,

[8] http://www.fakenewswatch.com/ http://www.propornot.com/p/the-list.html

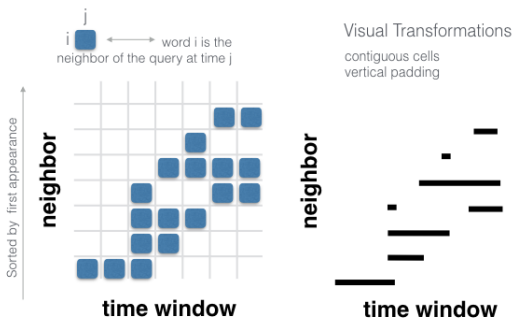


Figure 1: Our visual metaphor stems from an adjacency representation A of the nearest neighbors of the query term. The rows of the matrix correspond to nearest neighbors, and the columns correspond to time windows. The cell a_{ij} is filled if word i is a neighbor of the query term at time j. To this matrix to make the matrix more readable by the user, we apply a visual transformation.

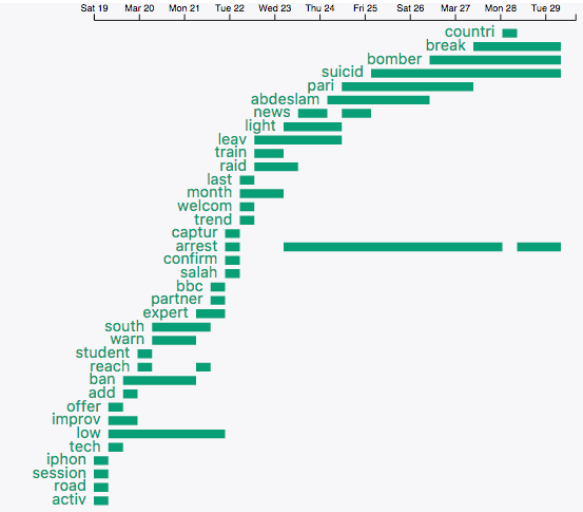
in our experience, many non-expert users are confused by the meaninglessness of the x- and y- coordinate space of the projected data, and have to be trained how to interpret such visualizations.

These problems are amplified when we consider dynamic data, where entities move throughout an embedding space over time. In our case, because embeddings are trained online, the meanings of the dimensions in the embeddings are changing, in addition to the words embedded therein. So, it is not correct to use traditional approaches to project an entities at different time points into the same space using the features directly.

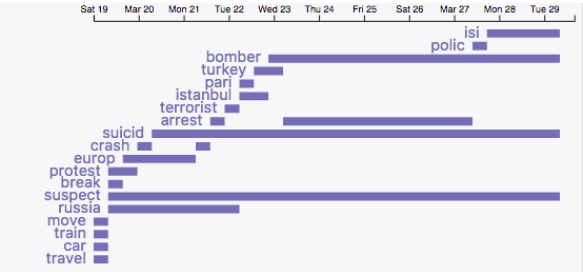
Our solution was to rely on a user driven querying and nearest neighbor technique to address these challenges. We allow users to query the embedding using a single keyword, as we assume the user has a few items of interest they wish to explore, and is not concerned with understanding the entire embedding. This allows us to frame our dynamic embedding visualization problem as a dynamic graph visualization problem (Beck et al., 2014), specifically visualizing dynamic ego-networks.

Our visual representation shows how the nearest neighbors of a user-provided query term change over time. The user can choose the k nearest neighbor words shown in the visualization. We encode time on the x-axis, whereas the y-axis is used to represent each nearest neighbor word returned by the query. This is a matrix representation of the nearest neighbors of the query term over time, as illustrated in Figure 1.

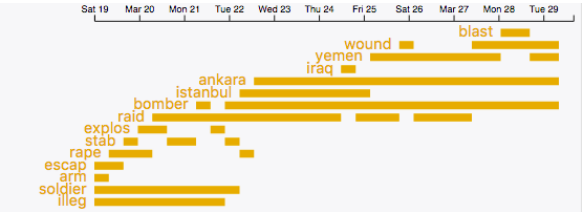
We apply a visual transformation to this matrix to make it easier to understand by replacing adjacent matrix cells with contiguous lines, and



(a) Belgium



(b) Germany



(c) United Kingdom

Figure 2: Visualization of dynamic embedding queries for the word "bomb" across the regions "Belgium," "Germany," and "United Kingdom" are shown. Time is encoded on the horizontal axis, and words are sorted by first occurrence (as a nearest neighbor) for the query term.

adding spacing between rows to help distinguish the query results. The words on the y-axis are sorted in the order they first become a neighbor of the query term. This helps the user see more recent terms, as they will float to the top, versus more persistent terms, which sink to the bottom, and have longer lines. Figure 2 shows a screenshot of our interface containing three of regional dynamic embeddings available for the term "bomb."

Users can compare visualizations of query results side by side in the interface, but we also designed a more explicit comparison of embeddings using a modified version of our visualization technique. Our goal for this comparison was to highlight similarities across two or more dynamic embedding queries over time. We accomplish this by first finding the shared neighbors of these queries within each time step, which is illustrated in Figure 3. We show the results of these queries using the same visual metaphor as described above with an additional embellishment. The thickness of the line at a given time now encodes the number of shared neighbors across the query results at that time. Also, when a query result is shared by more than one query in the combined chart, its corresponding line is filled black, otherwise it retains its original color corresponding to its region. Figure 4 shows an example of combining the query results for "bomb" across regions "Belgium," "Germany," and "United Kingdom."

4.2 Implementation

Our tool is a web application (i.e., client-server model) implemented using Python and Flask[9] for the server and React[10] and D3[11] for the client. The server is responsible for executing the query on the embeddings, whereas the client is responsible managing the users queries and visualizing the results. This separation of concerns means that the server assumes a large memory footprint[12] and processing burden, allowing the clients (i.e., web browsers) to be lightweight. This enables the interface to be used on a typical desktop or even a mobile device by multiple users simultaneously.

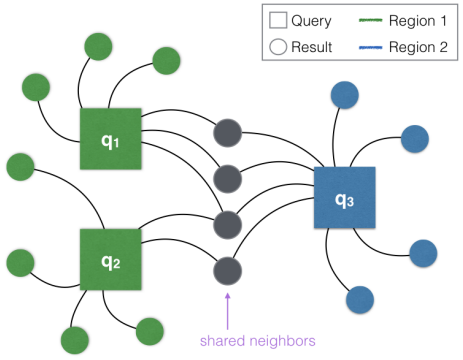


Figure 3: Dynamic embedding queries are combined by finding the shared neighbors across their query results at each time step. This example shows how three separate queries $\{q_1, q_2, q_3\}$ across two regions could have overlap in the result words within a single timestamp.

[9] http://flask.pocoo.org
[10] https://facebook.github.io/react/
[11] http://d3js.org
[12] For our Brussels data set, each dynamic embedding requires approximately 500MB of disk space and 2GB in memory after the data structures are created.

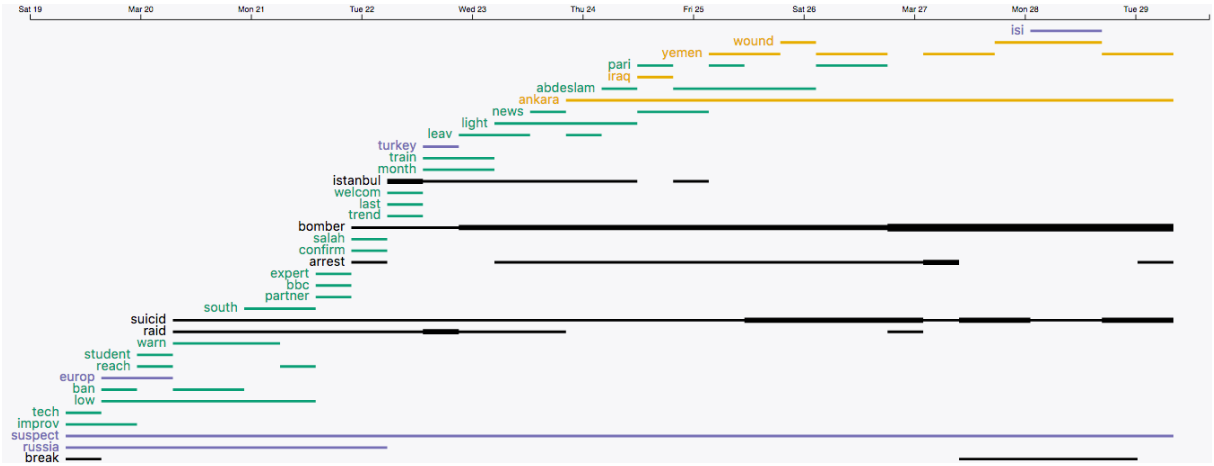


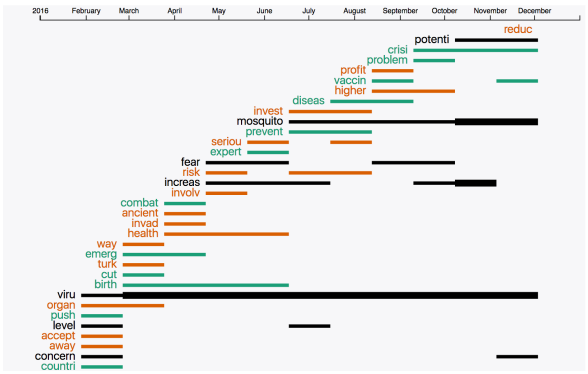
Figure 4: The dynamic embedding queries from Figure 2 are combined into a single chart to support a more explicit comparison of the dynamic embeddings across countries – Belgium (green), German (purple), UK (orange). Where the results overlap from the individual queries, a thicker black line is drawn.

Finding the k-nearest neighbors of a query term in the embedding could take a long time to query for dynamic embeddings with many dimensions and entities. We relied on the "ball tree" data structure available in scikit-learn[13] to help speed up the query. This data structure relies on the Euclidean distance metric, instead of cosine distance, which is considered a best practice. However, after spot checking a few relevant queries using cosine distance, we did not see a qualitative difference between the two metrics, and continued using the ball tree because of the performance advantage. One ball tree is computed for each region and time window, which has a large up front cost, but afterwards our tool provides embedding queries responsively (within 1 second per region). This approach is scalable because each query can divided independently into (region $\times$ time window) sub-tasks, allowing the overall calculation to be distributed easily in a map-reduce architecture.

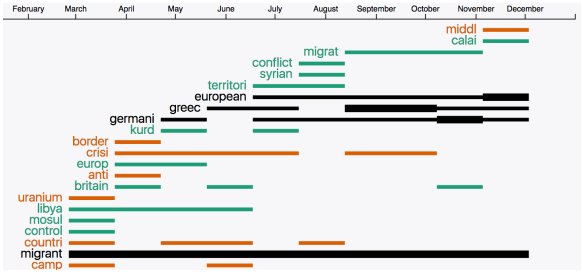
Analyzing Brussels Embeddings Figure 4 shows an example of combining the query results for "bomb" across regions "Belgium," "Germany," and "United Kingdom." We observe that the shared neighbors of the query word "bomb" are *Istanbul* (March 22 - 25), *suicide* (March 20 - 29), *arrest* (March 23 - 27), and *bomber* (March 22 - 29). The words *Paris* and *Abdeslam* are the neighbors only in Belgium, *wound*, *Yemen* and *Iraq* – in the UK, and *Europe, suspect* and *Russia* – in Germany.

[13] http://scikit-learn.org/stable/modules/generated/sklearn.neighbors.BallTree.html

Analyzing Suspicious News Embeddings Figure 5 shows the results for an example query word pairs: (a) "zika" and "risk" and (b) "Europe" and "refugee" learned from content extracted from suspicious and verified news in 2016. We found that *potential, mosquito, increase, virus* and *concern* are shared neighbors of two query words "zika" and "risk". We observed that *European, Greece, Germany* and *migrant* are shared neighbors of two query words "Europe" and "refugee".



(a) Zika and Risk



(b) Europe and Refugee

Figure 5: Visualization of dynamic embeddings for the words "zika" and "risk" with 2 neighbors learned from verified (green) and unverified (orange) news on Twitter.

5 Conclusion

We have presented ESTEEM, a novel framework for visualizing and qualitatively evaluating spatiotemporal embeddings learned from large amounts of dynamic text data. Our system allows users to explore specific aspects of text streaming corpus using continuous word representations. Unlike any other embedding visualization, our tool allows contrasting word representation differences over time across other dimensions e.g., geolocation, news types etc. For future work we plan to improve the tool by allowing the user to query using phrases and hashtags.

6 Acknowledgments

This research was conducted under the High-Performance Analytics Program at Pacific Northwest National Laboratory, a multiprogram national laboratory operated by Battelle for the U.S. Department of Energy. The authors would like to thank L. Phillips, J. Mendoza, K. Shaffer, J. Yea Jang and N. Hodas for their help with this work.

References

Fabian Beck, Michael Burch, Stephan Diehl, and Daniel Weiskopf. 2014. The state of the art in visualizing dynamic graphs. *EuroVis STAR* 2.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Proceedings of NIPS*. pages 4349–4357.

Jason Chuang, Daniel Ramage, Christopher Manning, and Jeffrey Heer. 2012. Interpretation and trust: Designing model-driven visualizations for text analysis. In *Proceedings of SIGCHI*. pages 443–452.

Anna Gladkova and Aleksandr Drozd. 2016. Intrinsic evaluations of word embeddings: What can we do better? *Proceedings of ACL* .

Fréderic Godin, Baptist Vandersmissen, Wesley De Neve, and Rik Van de Walle. 2015. Named entity recognition for twitter microposts using distributed word representations. In *Proceedings of ACL-IJCNLP*.

Kristina Gulordava and Marco Baroni. 2011. A distributional similarity approach to the detection of semantic change in the Google Books Ngram corpus. In *Proceedings of GEMS*. pages 67–71.

William Hamilton, Jure Leskovec, and Dan Jurafsky. 2016a. Diachronic word embeddings reveal statistical laws of semantic change. In *Proceedings of ACL*.

William L Hamilton, Jure Leskovec, and Dan Jurafsky. 2016b. Cultural shift or linguistic drift? comparing two computational measures of semantic change. In *Proceedings of EMNLP*.

Zellig S Harris. 1954. Distributional structure. *Word* 10(2-3):146–162.

Yoon Kim, Yi-I Chiu, Kentaro Hanaki, Darshan Hegde, and Slav Petrov. 2014. Temporal analysis of language through neural language models. *Proceedings of ACL* .

Tao Lei, Yuan Zhang, Regina Barzilay, and Tommi Jaakkola. 2014. Low-rank tensors for scoring dependency structures. In *Proceedings of ACL*.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proceedings of ACL*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. In *Proceedings of ICLR*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionally. In *Proceedings of NIPS*.

Alexandre Passos, Vineet Kumar, and Andrew McCallum. 2014. Lexicon infused phrase embeddings for named entity resolution. In *Proceedings of CoNLL*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of EMNLP*.

Cícero Nogueira Santos and Bianca Zadrozny. 2014. Learning character-level representations for part-of-speech tagging. In *Proceedings ICML*.

Tobias Schnabel, Igor Labutov, David Mimno, and Thorsten Joachims. 2015. Evaluation methods for unsupervised word embeddings. In *Proceedings of EMNLP*.

Daniel Smilkov, Nikhil Thorat, Charles Nicholson, Emily Reif, Fernanda B Viégas, and Martin Wattenberg. 2016. Embedding projector: Interactive visualization and interpretation of embeddings. *arXiv preprint arXiv:1611.05469* .

Ian Stewart, Dustin Arendt, Eric Bell, and Svitlana Volkova. 2017. Measuring, predicting and visualizing short-term change in word representation and usage in vkontakte social network. In *Proceedings of ICWSM*.

Laurens Van Der Maaten. 2014. Accelerating t-sne using tree-based algorithms. *Journal of machine learning research* 15(1):3221–3245.

Yating Zhang, Adam Jatowt, and Katsumi Tanaka. 2016. Towards understanding word embeddings: Automatically explaining similarity of terms. In *Proceedings of Big Data*. IEEE, pages 823–832.