Semantic Frame Embeddings for Detecting Relations between Software Requirements

Waad Alhoshan, Riza Batista-Navarro and Liping Zhao
School of Computer Science
University of Manchester
Oxford Road, Manchester M13 9PL, UK

Abstract

The early phases of requirements engineering (RE) deal with a vast amount of software requirements (i.e.,requirements that define characteristics of software systems), which are typically expressed in natural language. Analysing such unstructured requirements, usually obtained from stakeholders' inputs, is considered a challenging task due to the inherent ambiguity and inconsistency of natural language. To support such a task, methods based on natural language processing (NLP) can be employed. One of the more recent advances in NLP is the use of word embeddings for capturing contextual information, which can then be applied in word analogy tasks. In this paper, we describe a new resource, i.e., embedding-based representations of semantic frames in FrameNet, which was developed to support the detection of relations between software requirements. Our embeddings, which encapsulate contextual information at the semantic frame level, were trained on a large corpus of requirements (i.e., a collection of more than three million mobile application reviews). The similarity between these frame embeddings is then used as a basis for detecting semantic relatedness between software requirements. Compared with existing resources underpinned by frame embeddings built upon pre-trained vectors, our proposed frame embeddings obtained better performance against judgments of an RE expert. These encouraging results demonstrate the potential of the resource in supporting RE analysis tasks (e.g., traceability), which we plan to investigate as part of our immediate future work.

1 Introduction

As a part of Requirements Engineering (RE), requirements analysis is "a critical task in software development as it involves investigating and learning about the problem domain in order to develop a better understanding of stakeholders actual goals, needs, and expectations" (Hull et al., 2017). However, it is a challenge to analyse requirements to find relations between them, especially implicit ones, i.e., those that are not expressed explicitly and formally, especially within a lengthy document. As stated by Ferrari et al. (2017), these challenges are mainly due to the semantic ambiguity and incompleteness inherent to natural language. Moreover, performing an RE analysis task, e.g. by manually inspecting words and implicit or explicit relations between requirements, is a time-consuming and error-prone procedure (Fernández et al., 2017).

One of the approaches that has drawn the attention of the RE research community is semantic analysis. Representing under-specified meanings within requirements in a structured manner will lead to a more efficient way for conducting RE analysis task. As an example, Mahmoud and Niu (2015) discussed the importance of using techniques for measuring semantic relatedness in tracing links (or relations) between requirements. This mimics the human mental model in understanding links between pieces of text through their implicit meanings. Natural language processing (NLP) tools and techniques offer viable solutions to many tasks in RE, including requirements analysis (Dalpiaz et al., 2018). However, the majority of the available NLP techniques and resources are not domain-specific, i.e., they are trained or

built based on general-domain data sets (e.g., news articles). For this reason, a recent research direction in RE calls for "customizing general NLP techniques to make them applicable for solving the problems requirements engineers face in their daily practice" (Ferrari et al., 2017).

In this work, we present a new resource, i.e., semantic frame embeddings, built upon semantic frames in FrameNet (Baker et al., 1998). To demonstrate an application to requirements analysis, we employed our semantic frame embeddings in computing semantic relatedness between software requirements at a semantic frame level.

The rest of this paper is organised as follows: Section 2 provides background information on FrameNet and word embeddings, while Section 3 presents the method we carried out to generate the frame embeddings. In Section 4, we discuss the results of employing the obtained frame embeddings in a semantic relatedness measurement task. Finally, we conclude and briefly discuss our ongoing work in Section 5.

2 Background

2.1 A Brief Overview on FrameNet

Fillmore (1976) proposed the linguistic theory of semantic frames, stating that each word in a language is accompanied by essential knowledge which is important to understand its full meaning. For example, words such as "store" and "keep" are usually accompanied by the following elements: (1) an agent that performs a storing event; (2) an object which is a result of the storing event; and (3) the location where the object is kept.

FrameNet[1] is a web-based general-domain semantic lexicon that implements the semantic frame theory. Initially started by Baker et al. (1998), it has continued to grow and now contains more than 1,200 semantic frames (Baker, 2017). For every semantic frame in FrameNet, the following information is given: frame title, definition, frame elements and lexical units (LUs). LUs are words that evoke the frame, represented as a combination of their lemmatised form and part-of-speech (POS) tag. The concept of keeping an object, for example, which is stored in FrameNet as a semantic frame entitled *Storing* is evoked by the LU *save.v* where *v* stands for verb, among other LUs. Its core frame elements, which are essential in understanding the meaning of the frame, include Agent, Location and Theme. FrameNet also catalogues non-core frame elements which are used to enhance the understanding of a specific frame. For the *Storing* frame, Manner, Duration, and Explanation are considered as non-core elements.

In Figure 1, we demonstrate the use of semantic frames and their related LUs for representing a set of software requirements. From the given example in Figure 1, we can identify the requirements and conditions for implementing the designated system, e.g., accessing restrictions to the documents as shown in in Req-1 and Req-2. The need to update records on a regular basis as described in Req-3 and Req-4 are also shown. These requirements are abstractly represented by using FrameNet frames. For instance, accessing restrictions are represented by the *Deny_or_grant_permission* and *Preventing_or_letting* frames from FrameNet. Similarly, the processed materials "reports", "logs", and "contact information" are captured by the *Text*, *Records* and *Information* frames, respectively.

Furthermore, some frames (e.g., *Storing*, *Records*, *Verification*, and *Frequency*) are repeated amongst the requirements in Figure 1, boosting the semantic relatedness between these requirements. FrameNet holds a representation of semantic relations between its frames (Baker, 2017). For example, the frame *Record* inherits from the *Text* frame. Using such semantic relations could help create links between annotated requirements, as reported in our previous work Alhoshan et al. (2018c). However, according to Baker (2017) not all frames in FrameNet are semantically connected. For example, *Information* and *Records* are not linked in any way. Similarly, *Deny_or_grant_permission* and *Prenveting_or_letting* are not connected in FrameNet although both frames share some LUs (e.g., *permit.v*). For this reason, rather than rely on the semantic relations encoded in FrameNet, we sought another way to find semantic links between FrameNet frames.

[1] https://framenet.icsi.berkeley.edu/

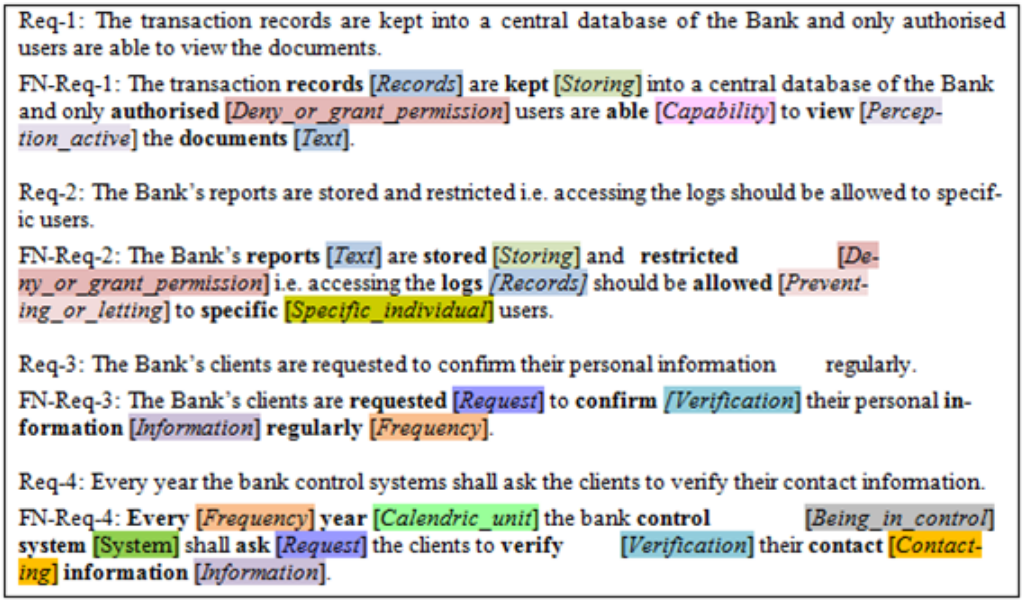

Figure 1: A set of software requirements, where "Req" refers to the raw requirements and "FN-Req" refers to the requirements annotated with FrameNet frames titles (highlighted with colours) and their evoked LUs (in bold font).

2.2 Word Embeddings

One of the recent advances in NLP research is the use of word embeddings as a method for capturing the context of any given word in a corpus of documents. According to Mikolov et al. (2013), word embeddings allow words with similar, or related meanings, to have similar vector representations. They are learned based on the principle of distributional semantics, which posits that words occurring in the same context have similar or related meanings (Harris, 1954). Deep learning offers a framework for representing word context as real-valued vectors, that goes beyond the counting of co-occurrences and takes into account word order as well (Bengio et al., 2003). For training word embeddings, a large and representative corpus is needed. There are existing pre-trained, general-domain word embeddings ready for use, e.g., the Word2Vec embeddings trained on 100 billion words from Google News (Mikolov et al., 2013).

In general, word embeddings have helped boost the performance of various NLP tasks. An example is word analagy, where word embeddings provide the capability to calculate semantic similarities between words (Fu et al., 2014). However, the use of word embeddings can lead to even better performance if they are trained on corpora specific to the domain of interest or application. This could potentially reduce the problem of out-of-vocabulary (OOV) words (Jozefowicz et al., 2016), i.e., the lack or sparsity of instances of certain words in the training corpus, which leads to not being able to capture or map their context in embedding vectors. The solution to such cases is typically based on simply ignoring the OOV words, which is not ideal.

In this work, we proposed a solution for mitigating text sparsity that is based on semantic frames. Rather than mapping each word in the text, we target a group of words which represent a semantic frame, hence producing *semantic frame embeddings*. There are previously reported efforts that proposed the use of frame embeddings, e.g., Sikos and Padó (2018) and Alhoshan et al. (2018c). In our work, we aim to develop frame embeddings that are suitable for captruing the context of RE-related documents.

3 Semantic Frame Embeddings

Our method for generating frame embeddings was previously discussed in our prior work Alhoshan et al. (2018c) which we employed existing word embeddings developed by Efstathiou et al. (2018). In this paper, we trained our own word embeddings which then formed the basis for generating semantic frame embeddings. Afterwards, we measured the semantic relatedness between frames using different similarity metrics. Finally, we selected the most suitable metric for applying frame embeddings to the RE domain.

3.1 Preparation of Training Data

As a first step, we generated a corpus of requirements documents that are more similar to software requirements, i.e., a collection of user reviews of mobile applications. Using the web-based AppFollow tool[2], reviews from different mobile application repositories (e.g., Apple Store and Google Play) were retrieved. The user reviews covered different categories of mobile applications, i.e., business, sports, health, travel, technology, security, games, music, photos, videos, shopping, lifestyle, books, social networking, finance. While each review came with metadata such as review date, title and per-user application rating, we took into consideration only the textual content of the reviews. This resulted in a total of 3,019,385 unique reviews/documents in our training data set.

The documents in the training data set were then preprocessed with the following steps: sentence splitting, tokenisation, stop-word removal, part-of-speech (POS) tagging and lemmatisation. The preprocessing results allowed us to automatically check for the occurrence of LUs (associated with semantic frames) catalogued in FrameNet, in order to assess the data set's coverage of semantic frames. Based on this, we were able to determine that our mobile application reviews data set covers all of the 123 semantic frames annotated in FN-RE corpus (a FrameNet annotated corpus of software requirements presented in Alhoshan et al. (2018a,b).

3.2 Training Word Embeddings

Utilising the preprocessed mobile application reviews data set as a corpus, we trained word embeddings using the continuous bag-of-words (CBOW) learning method of *Word2Vec* as proposed by Mikolov et al. (2013). A word embedding vector was trained for each LU, which was represented as a combination of its lemmatised form and POS tag. Taking into account the POS tag of an LU makes it possible to train different vectors for words with the same lemma but different parts of speech. It is preferable, for example, to train a vector for "form" as a verb (*form.v*) that is different from the vector for "form" as a noun (*form.n*). The size of each vector was set to 300, following previously reported work in Sikos and Padó (2018) and Mikolov et al. (2013).

3.3 Generating Frame Embeddings

The word embedding vectors resulting from the previous step were then used to form an embedding-based representation of semantic frames, i.e., *frame embeddings*. That is, for any given semantic frame F, we collected the vectors corresponding to the LUs that evoke it. The average of these LU vectors is then computed and taken as the frame embedding for F. For instance, as 11 LUs are associated with the *Creating* frame in FrameNet, a vector containing the average over the 11 word embedding vectors corresponding to these LUs was obtained as part of this step.

3.4 Measuring Frame-to-Frame Semantic Relatedness

The generated frame embeddings were employed in computing relatedness between semantic frames. Following our method described in Alhoshan et al. (2018c), we used the cosine similarity metric. For

[2]https://appfollow.io

FrameNet frames *X* and *Y*, let *FR(X,Y)* denote the relatedness between these two frames:

$$FR_{Cosine}(X,Y) = \frac{\mathbf{F_X} \cdot \mathbf{F_Y}}{\|\mathbf{F_X}\|\|\mathbf{F_Y}\|} \tag{1}$$

where $\mathbf{F_X}$ and $\mathbf{F_Y}$ are the frame embedding vectors for *X* and *Y*, respectively.

The cosine similarity metric measures the angle between two vectors (i.e., frame embeddings). If the vectors are close to parallel (e.g., with *R(X,Y)* $\approx$ 1) then we consider the frames as similar, whereas if the vectors are orthogonal (i.e., with *R(X,Y)* $\approx$ 0), then we can say that the frames are not related.
In addition, we used two other similarity metrics, Euclidean Distance and Manhattan Distance, for later comparison:

$$FR_{Euclidean}(X,Y) = \sqrt{(\mathbf{Fx} - \mathbf{Fy})^{\mathbf{2}}} \tag{2}$$

$$FR_{Manhattan}(X,Y) = \|\mathbf{Fx} - \mathbf{Fy}\| \tag{3}$$

Similar to the cosine metric, the Euclidean and Manhattan metrics measure the distance between two data points (i.e., distance between the two frame embeddings) to detect their similarity—i.e., if the data points are close together (with a shorter distance), this is considered as a higher similarity between the designated frame embeddings to be measured.

The Manhattan distance metric calculates the path between any two data points as it would be placed in a grid-like path, whereas the Euclidean distance measures the distance as a straight-line.

An issue that is related to the distance scores of both Euclidean and Manhattan metrics is that the results can be too large (i.e., greater than 1) if the data points to be compared are sparse. For this reason, we applied the Z_{score} in order to normalise obtained results from Euclidean and Manhattan distance metrics separately:

$$Z_{score}(Fx,Fy) = \frac{Dxy - \mu}{\alpha} \tag{4}$$

Z_{score} is a function for normalising Dxy which is the similarity distance calculated by FR between *Fx* and *Fy* (calculated using either Euclidean or Manhattan distance) where μ is the mean distance over all frame pairs, and α is the standard deviation.

For implementing the methods described above, we employed various Python-based packages. The preprocessing pipeline was implemented using the NLTK Python package[3] as well as NodeBox[4]. Meanwhile, the Word2Vec implementation available in the Gensim package[5] facilitated the training of word embeddings. The numpy package[6] was used in generating the frame embeddings and calculating similarity scores, and matplotlib [7] for visualising the frame embeddings relations.

4 Results

In this section, we discuss the results obtained by using the frame embeddings generated by the method described above. We used the 123 semantic frames in FrameNet that are annotated in the FN-RE corpus, reported in Alhoshan et al. (2018b). The first author of this paper, who is a PhD candidate investigating the use of NLP techniques in RE, annotated the semantic relatedness between the selected frames pairs as "yes" if the frame pair is semantically related according to their definition and related (or shared) LUs, and "no" otherwise.

We applied the three similarity metrics discussed in Section 3.4, on the frame embeddings of the selected frame pairs as exemplified in Table 1. The results obtained from Euclidean and Manhattan

[3]https://www.nltk.org/
[4]https://www.nodebox.net/code/index.php/Linguistics
[5]https://radimrehurek.com/gensim/models/word2vec.html
[6]http://www.numpy.org/
[7]https://matplotlib.org/

distance metrics are normalised according to our discussion above. We considered 0.50 as a threshold value to indicate semantic relatedness for any frame pair in the set, following prior work by Alhoshan et al. (2018c). From the given results in Table 1, it is clear that using the cosine metric provides more reliable relatedness scores that are close to the registered human-judgement—i.e., out of six positive scores of semantic relatedness on the given frame pairs shown in Table 1, the cosine metric identified five of them as semantically related with scores that are equal or higher than the used minimum threshold value. The cosine metric is generally used to identify semantic relatedness regardless of the magnitude of the frame embedding, whereas both the Euclidean and Manhattan metric measure the actual magnitude distance between the frame embeddings. For example, the frame *Sending* occurs 0.824% in the training corpus and the frame *Receiving* occurs only 0.007% of the time. The Euclidean and Manhattan metrics measure the similarity of these two frames depending on how often they occurred in the corpus, whereas cosine similarity measures only the angle of their vector representations. For this reason, we selected the cosine metric to compare our frame embeddings with the pre-trained frame embeddings.

Table 1: Results of frame pair semantic relatedness scores according to the applied similarity metrics. Underlined values pertain to the highest valued score (above the minimum threshold) for each semantically related frame pair.

Frames Pairs	**Human-judgements: Semantically related?**	**Euclidean Distance (Normalised)**	**Manhattan Distance (Normalised)**	**Cosine similarity**
(Sending, Creating)	**Yes**	0.8622	-0.6653	0.5794
(Sending, Intentionally_create)	**Yes**	-0.6249	-0.62574	0.5383
(Sending, Receiving)	**Yes**	-0.3409	-0.3443	0.5356
(Sending, Recording)	No	-0.2469	-0.2050	0.4538
(Creating, Intentionally_create)	**Yes**	-1.3837	-1.3967	0.9034
(Creating, Receiving)	No	-0.3098	-0.3406	0.4697
(Creating, Recording)	No	-0.3329	-0.3422	0.4573
(Intentionally_create, Receiving)	No	-0.2057	-0.2170	0.3703
(Intentionally_create, Recording)	Yes	-0.2570	-0.2769	0.3778
(Receiving, Recording)	**Yes**	-0.1992	-0.2102	0.5081

In Table 2, we compare the characteristics of the frame embeddings based on word embeddings pre-trained on news articles as used by Sikos and Padó (2018) (Column A), those pre-trained on Stackoverflow posts by Efstathiou et al. (2018) used in Alhoshan et al. (2018c) (Column B), and our own proposed embeddings (Column C).

Table 2: Comparison between our proposed Frame Embeddings (C) and the two available frame embeddings (A) and (B).

Feature	**FrameNet Corpus Frame Embedding (A)**	**FN-RE Corpus Frame Embedding version 1.0 (B)**	**FN-RE Corpus Frame Embedding version 2.0 (C)**
Trained data set size	31.0 MB	1.5 GB	990.1 MB
data set context	News articles	Stack overflow technical posts User	reviews of mobile applications
Number of words entries	21,121 words	1.7 million words	1.6 million words
Language Model	Word2Vec (dimension size: 300)	Word2Vec (dimension size: 200)	Word2Vec (dimension size: 300)
Context	General	Software Engineering	Requirements Engineering

The frame embeddings (A) and (B) are compared with our proposed frame embeddings (C) based on a data set of frame pairs whose semantic relatedness has been labelled. The results are shown in Table 3. For example, *Creating* and *Intentionally_create* frames, have some LUs in common (e.g., *create.v*, *generate.v* and *make.v*). Both frames are connected via the inheritance relation *is-a* in FrameNet.

During the annotation of the FN-RE corpus, described in Alhoshan et al. (2018a) and Alhoshan et al. (2018b), those two frames (*Creating* and *Intentionally_create*) in particular are overlapping and describe very similar contexts. As shown in Table 3, the frame pair *(Creating, Intentionally_create)* obtained a significant relatedness score of 0.903 according to our frame embeddings (C). More importantly, our frame embeddings provided overall semantic relatedness results that are closer to our judgement, as we discussed previously in this section. Such encouraging results indicate that using a training corpus that is specific to the RE context provides improved results.

Table 3: Semantic relatedness scores (computed using cosine similarity) for each frame pair according to our proposed frame embeddings (C) and the two other frame emebeddings (A) and (B). Underlined values pertain to the highest score (above the minimum threshold) for each semantically related frame pair.

Compared Frames Pairs	**Human-judgments: Semantically related?**	**General FrameNet Corpus Frame (A)**	**FN-RE Corpus Frame Embedding version 1.0 (B)**	**FN-RE Corpus Frame Embedding version 2.0 (C)**
(Sending, Creating)	**Yes**	0.2642	0.3722	0.5795
(Sending, Intentionally_create)	**Yes**	0.2320	0.4175	0.5383
(Sending, Receiving)	**Yes**	0.2605	0.6466	0.5356
(Sending, Recording)	No	0.2356	0.5587	0.4538
(Creating, Intentionally_create)	**Yes**	0.8318	0.4338	0.9034
(Creating, Receiving)	No	0.3433	0.2677	0.4697
(Creating, Recording)	No	0.3084	0.2508	0.4573
(Intentionally_create, Receiving)	No	0.3008	0.3496	0.3703
(Intentionally_create, Recording)	Yes	0.3620	0.2867	0.3778
(Receiving, Recording)	**Yes**	0.2722	0.2875	0.5081

As shown in previous work, semantic frames are a promising means for capturing the meaning of software requirements ,e.g., Alhoshan et al. (2018c). Our encouraging results demonstrate that with careful selection of a similarity metric (for measuring semantic relatedness) and a suitable training data set representing software requirements, our proposed semantic resource (i.e., the frame embeddings) combines the strengths of semantic frames and embedding-based representations– which can be integrated with RE tools to support the task of software requirements analysis and traceability.

5 Conclusion

We presented a novel language resource to aid in finding semantic relations between software requirements, in support of RE tasks. The proposed resource is based on the development of an embedding-based representation of semantic frames in FrameNet (i.e., frame embeddings), trained on a large corpus of user requirements, consisting of more than three million mobile application reviews. In our immediate future work, we shall integrate this resource with RE methods for analysing and tracing semantic relatedness of software requirements. This in return, will aid in organising and grouping related system features described in requirements documents.The frame embeddings are publicly available at `https://doi.org/10.5281/zenodo.2605273`.

References

Alhoshan, W., Batista-navarro, R., and Zhao, L. (2018a). A framenet-based approach for annotating software requirements. In Torrent, T. T., Borin, L., and Baker, C. F., editors, *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Paris, France. European Language Resources Association (ELRA).

Alhoshan, W., Batista-Navarro, R., and Zhao, L. (2018b). Towards a corpus of requirements documents enriched with semantic frame annotations. In *2018 IEEE 26th International Requirements Engineering Conference (RE)*, pages 428–431.

Alhoshan, W., Zhao, L., and Batista-Navarro, R. (2018c). Using semantic frames to identify related textual requirements: An initial validation. In *Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, ESEM '18, pages 58:1–58:2, New York, NY, USA. ACM.

Baker, C. F. (2017). Framenet: Frame semantic annotation in practice. In *Handbook of Linguistic Annotation*, pages 771–811. Springer.

Baker, C. F., Fillmore, C. J., and Lowe, J. B. (1998). The berkeley framenet project. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pages 86–90. Association for Computational Linguistics.

Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. (2003). A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155.

Dalpiaz, F., Ferrari, A., Franch, X., and Palomares, C. (2018). Natural language processing for requirements engineering: The best is yet to come. *IEEE Software*, 35(5):115–119.

Efstathiou, V., Chatzilenas, C., and Spinellis, D. (2018). Word embeddings for the software engineering domain. In *Proceedings of the 15th International Conference on Mining Software Repositories*, pages 38–41. ACM.

Fernández, D. M., Wagner, S., Kalinowski, M., Felderer, M., Mafra, P., Vetrò, A., Conte, T., Christiansson, M.-T., Greer, D., Lassenius, C., et al. (2017). Naming the pain in requirements engineering. *Empirical software engineering*, 22(5):2298–2338.

Ferrari, A., DellOrletta, F., Esuli, A., Gervasi, V., and Gnesi, S. (2017). Natural language requirements processing: a 4d vision. *IEEE Software*, (6):28–35.

Fillmore, C. J. (1976). Frame semantics and the nature of language. *Annals of the New York Academy of Sciences*, 280(1):20–32.

Fu, R., Guo, J., Qin, B., Che, W., Wang, H., and Liu, T. (2014). Learning semantic hierarchies via word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1199–1209.

Harris, Z. S. (1954). Distributional structure. *¡i¿WORD¡/i¿*, 10(2-3):146–162.

Hull, E., Jackson, K., and Dick, J. (2017). *Requirmenets Engineering*. Springer, London.

Jozefowicz, R., Vinyals, O., Schuster, M., Shazeer, N., and Wu, Y. (2016). Exploring the limits of language modeling. *arXiv preprint arXiv:1602.02410*.

Mahmoud, A. and Niu, N. (2015). On the role of semantics in automated requirements tracing. *Requir. Eng.*, 20(3):281–300.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Sikos, J. and Padó, S. (2018). Using embeddings to compare framenet frames across languages. In *Proceedings of the First Workshop on Linguistic Resources for Natural Language Processing*, pages 91–101.