Neural GRANNy at SemEval-2019 Task 2: A combined approach for better modeling of semantic relationships in semantic frame induction

Nikolay Arefyev[1,2], **Boris Sheludko**[1,2], **Adis Davletov**[1,2], **Dmitry Kharchev**[1,2], **Alex Nevidomsky**[1], and **Alexander Panchenko**[3,4]

[1]Samsung R&D Institute Russia, Moscow, Russia
[2]Lomonosov Moscow State University, Moscow, Russia
[3]Skolkovo Institute of Science and Technology, Moscow, Russia
[4]Language Technology Group, University of Hamburg, Hamburg, Germany

Abstract

We describe our solutions for semantic frame and role induction subtasks of SemEval 2019 Task 2. Our approaches got the highest scores, and the solution for the frame induction problem officially took the first place. The main contributions of this paper are related to the semantic frame induction problem. We propose a combined approach that employs two different types of vector representations: dense representations from hidden layers of a masked language model, and sparse representations based on substitutes for the target word in the context. The first one better groups synonyms, the second one is better at disambiguating homonyms. Extending the context to include nearby sentences improves the results in both cases. New Hearst-like patterns for verbs are introduced that prove to be effective for frame induction. Finally, we propose an approach to selecting the number of clusters in agglomerative clustering.

1 Introduction

Semeval-2019 Task 2 consisted of three subtasks, this paper presents solutions to all three which were all performing better than other submitted approaches. The first solution officially took the first place in the competition, the other two used tuning on the development set provided by the organizers, which was then interpreted as using additional corpora.

Semantic Frame Induction (Subtask A) is the task of grouping target word occurrences in a text corpus according to their frame (meaning and semantic arguments structure). Target words are usually verbs, nouns, and adjectives (these have argument structure; however in the shared task dataset only verbs were present). For instance, the verbs *rise, fall* and *climb* in the sentences *The dollar is rising, which makes Russian economy unstable* and *The dollar fell 1% in September after climbing 2% in August* should be clustered together, while the verb *climb* in sentences like *People climb mountains* should be clustered separately. For the sake of brevity, occurrences of different words sharing the same frame will be called synonyms, and occurrences of the same word belonging to different frames will be called homonyms. This may violate the traditional meaning of these terms. For instance, *fall* and *rise* are not considered synonyms in the classical sense. Semantic Role Induction refers to finding realizations of semantic arguments in text and relating them to corresponding semantic frame slots. Generic role induction (subtask B.2) requires a small number of frame-independent roles like *Agent, Patient, Theme*, etc. Frame-specific role induction (subtask B.1) allows labeling arguments of each frame independently from other frames. For instance, *Microsoft* in *Microsoft bought Github* and *Google* in *Google opened new offices* should be labeled as the same role in B.2 but may be labeled differently in B.1. For further details please refer to QasemiZadeh et al. (2019).

In this paper, we focused mainly on the Frame Induction subtask. The main contributions for this subtask are the following. A combined approach to semantic frame induction is introduced, which clusters dense representations obtained from hidden layers of a masked LM first and sparse bag-of-words representations of possible substitutes for a word in context afterward. This approach resulted in better clustering of both synonyms and homonyms[1]. New Hearst-like patterns designed specifically for verbs were used and they proved to be beneficial for Semantic Frame Induction. Also, a simple but effective semi-supervised approach to selecting the number of clusters for agglomerative clustering was proposed. Finally, we proposed ex-

[1]GRANNy in the team name stands for General Relation Acquisition with Neural Networks

Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019), pages 31–38
Minneapolis, Minnesota, USA, June 6–7, 2019.

tending context with neighboring sentences which have shown consistent improvements for both of our representations. For solving subtask B.2 we used a semi-supervised approach of training logistic regression over features that were partly designed and partly learned in an unsupervised fashion. To ensure the best performance on verbs that were not present in training data (the majority of examples in the test) we used cross-validation with a lexical split, to select optimal features and hyperparameters. For solving subtask B.1 we trivially reused labels from B.2

2 Related Work

This section describes previous work which our approach is based on. Word Sense Induction (WSI) is the task of clustering occurrences of an ambiguous word according to their meaning which is similar to Frame Induction. One of the major differences from Frame Induction is that WSI doesn't require grouping together different words with similar meanings, however, we adopt some ideas from WSI in this work. Instead of graph or vector representation of word co-occurrence information traditionally used to solve WSI task, Baskaya et al. (2013) proposed exploiting n-gram language model (LM) to generate possible substitutes for an ambiguous word in a particular context. Their approach was one of the best in SemEval-2013 WSI shared task (Jurgens and Klapaftis, 2013). Struyanskiy and Arefyev (2018) proposed pretraining SOTA neural machine translation model built from Transformer blocks (Vaswani et al., 2017) to restore target words hidden from its input (replaced with a special token CENTERWORD). After pretraining, they exploited both predicted output embeddings to represent ambiguous words and attention weights to better weigh relevant context words in word2vec weighted average representation. A combination of these representations achieved SOTA results on one of the datasets from RUSSE'2018 Word Sense Induction for the Russian language shared task (Panchenko et al., 2018). Amrami and Goldberg (2018) develop ideas from Baskaya et al. (2013) exploiting neural bidirectional LM ELMO (Peters et al., 2018) instead of n-gram LM for generating substitutes. To improve results further they propose using dynamic symmetric patterns "T and _", "_ and T" (here "T" stands for the target word and "_" for the position at which we collect LM predictions). For instance, to represent the word *orange* in *He wears an orange shirt* instead of predicting what comes after *wears* in *He wears* they predict what comes after *and* in *He wears orange and* (similarly, for backward LM they predict what comes before *and* in *and orange shirt*). This provides more information to the LM because we don't hide the ambiguous word and forces it to produce its co-hyponyms instead of all possible continuations given a one-sided context. Other important contributions include lemmatizing substitutes to remove grammatical bias from representations (which was especially important for verbs) and using IDF weights to penalize frequent substitutes, which are probably worse for discriminating between senses. They achieve SOTA results on the SemEval-2013 WSI dataset.

Devlin et al. (2018) proposed BERT (Bidirectional Encoder Representations from Transformers). Like the model from Struyanskiy and Arefyev (2018), BERT is a deep NN built from Transformer blocks and pretrained on the task of restoring words hidden from its input (replaced with a special token [MASK], hence they named it masked LM). However they used much deeper models, pretrained them on much more data and predicted hidden words at each timestep rather than generating them as an output sequence. Also additional next sentence prediction task was used to pretrain the model for sentence pairs classification (like paraphrase detection and NLI). BERT has shown better results than previous SOTA models on a wide spectrum of natural language processing tasks.

3 Semantic Frame Induction

In this section, we describe our approaches to building vector representations of an occurrence of the target word (which is always a verb in the SemEval-2019 Frame Induction task dataset). The first approach exploits dense vector representations of the target word in a context obtained from hidden layers of BERT model. Another approach builds sparse TF-IDF BOW vectors from substitutes generated for the target word by BERT masked LM. We found that each model has its own downsides when used with non-trainable distance functions like cosine and Euclidean, and with traditional clustering algorithms like agglomerative clustering, DBScan, and affinity propagation. The

first approach didn't discriminate different senses of the same verb, the second one had problems with clustering together similar senses of different verbs. In preliminary experiments, we tried fixing the first problem by learning a distance function instead of using a fixed one, but this didn't help, presumably due to a very small amount of labeled data provided and restrictions on using additional labeled data. So our best performing algorithm is two-stage: it groups examples to a relatively small number of large clusters using the first representation (merging synonyms together while not taking into consideration homonyms) and then splits each of them into smaller clusters using the second representation (disambiguating homonyms). Finally, we describe our approach to clustering these vector representations and propose a technique for selecting the appropriate number of clusters.

3.1 BERT Hidden Representations

In the preliminary experiments, we compared dense representations from different layers of two BERT models pretrained on English texts: bert-base-uncased and bert-large-uncased with 3x more weights. While being significantly slower, the large model didn't show better clustering results for the development set, so we stuck to the base model. Presumably, fine-tuning the large model to the final task could reveal its superiority, but this would require much more labeled data that was provided. Interestingly, a weighted average of word2vec embeddings for context words proposed for WSI in Arefyev et al. (2018) showed similar results, which also supports the hypothesis that distance functions like cosine or Euclidean are not appropriate for BERT hidden representations. BERT-base consists of 12 Transformer blocks with 12 attention heads each, hidden state dimensionality is 768. It was pre-trained on lowercased texts split into subword units. Hyperparameters were selected on the development set, the best results were achieved using outputs of the layer 6 at timestep when the first subword of the verb was fed in. Also, better results were achieved when input texts were lemmatized. This can be explained by the large grammatical bias of LMs also noticed by Amrami and Goldberg (2018): it is much easier to correctly predict grammatical attributes like number, gender, tense from contexts, so it is more beneficial to assign higher probabilities to all verbs with correct tense than to all verbs with correct meaning when losses like cross-entropy are used, which results in large distance between occurrences of the same verb in the same meaning, but in different tenses.

3.2 Substitutes Representations

We adopt ideas from Amrami and Goldberg (2018) for our second approach to Frame Induction, with several important differences. First, we propose new patterns which are more suitable for verbs. Secondly, we use BERT, which proved to be better than ELMO for generating substitutes in our series of preliminary experiments. This is likely due to the fact that BERT takes into account the whole context in all of its layers, unlike bidirectional LM in ELMO, which consists of two independently trained language models, one using only right context, and another only left context. Lastly, we do hard clustering instead of soft clustering required for SemEval-2013 WSI, hence we do not sample from distributions predicted by LM, but instead, take the topmost probable substitutes. We found this approach works better than one doing soft clustering and then selecting the most probable cluster for each example.

To generate substitutes, a masked LM based on the bert-base-uncased model was utilized. It is likely that the large model could generate better substitutes, but we left it for future work. Non-lemmatized lowercased text was passed through all the layers of the model. We didn't add biases of the last linear layer to obtain less frequent but more contextually suitable subwords. We took K most probable substitutes to represent each example (K=40 was selected on the development set), lemmatized them to get rid of grammatical bias, and then built TF-IDF bag-of-words vectors. To improve results we employ symmetric patterns. Symmetric patterns were first proposed in Hearst (1992) and then used in many cases, including Widdows and Dorow (2002), Panchenko et al. (2012), Schwartz et al. (2015), to extract lexical relations like hyponymy, hypernymy, co-hyponymy, etc. from texts, and to augment lexical resources. However, we were not aware of any Hearst-like patterns designed specifically for verbs. Along with "T and _" pattern and trivial "T" and "_" patterns we proposed and experimented with "T and then _", "T and will _" and "T and then will _" patterns. We suppose that the meaning of a verb is better described not by its hypernyms or co-

hyponyms (which are traditionally extracted for nouns using patterns like "_ such as T" or "T and _") but rather by preceding and following events which are better extracted by the proposed patterns. "T and then _" pattern has shown the best results both for the development and the test sets. For instance, to generate substitutes for the verb *build* in *They are building phones* we pass *They are building and then* [MASK] *phones* and collect predictions at the masked timestep. We found that among others, substitutes like *export, distribute, ship* are generated for *Manufacturing* frame and *establish, open, close* for *Building* frame of the verb *build* allowing to discriminate between them. See Appendix A for examples.

3.3 Clustering

We experimented with K-means, DBScan, Affinity Propagation and Agglomerative clustering algorithms implemented in the scikit-learn (Pedregosa et al., 2011) and found agglomerative clustering to achieve the best results. To select hyperparameters of Agglomerative clustering for dense representations (number of clusters and distance functions between points and clusters) we used a simple yet effective semi-supervised approach: merge the development and test sets (labeled and unlabeled respectively) and perform grid search for hyperparameters that provide clustering with optimal value of the target metric (BCubed-f1 in our case) on the labeled subset. Almost always optimal results were obtained using cosine distance for points and average linkage for clusters (average distance between elements).

3.4 Combined Approach

Our best performing submission was made of a combination of techniques described above. At phase 1, we clustered dense representations using proposed semi-supervised agglomerative clustering. At phase 2, we split each cluster separately using sparse representations and conventional agglomerative clustering with cosine distance and average linkage (selected on the development set). We didn't use the semi-supervised tuning again because at that stage most clusters didn't contain labeled examples. During the blind evaluation period, we simply split each cluster into two (this method is denoted as **Combined** below). In the post-evaluation period, we experimented with more sophisticated approaches. Finally, our best results (denoted as *Combined2*) were obtained when the number of clusters at phase 2 was selected using silhouette score and small clusters (with less than 20 examples) or clusters with different target verbs were left intact. Also, during the post-evaluation period, we tried extending the context with nearby sentences (sentences with adjacent IDs in the Penn Treebank corpus). This allowed us to incorporate more information about the preceding and following events, which resulted in improved performance of both representations. In **Combined2** we passed a large context of maximum 7 sentences to the left and to the right for dense, and smaller context of 2 sentences on both sides for sparse representations (selected on the development set).

3.5 Dataset and Experiments

Due to limitations imposed by the task, we restricted ourselves to only using labeled data provided by the organizers. For the majority of our experiments, we used the development set that consisted of 600 examples of 35 verbs clustered into 41 frames. There are many examples of synonymy in this dataset but not so many of homonymy. Almost all ambiguous verbs have less than 5 examples for all frames except their most frequent frame, hence we used only verbs *join* and *believe* (54/9 and 12/8 examples of their first/second most frequent frame respectively) to select hyperparameters likely resulting in a suboptimal performance on the test.

For internal evaluation of different representations and hyperparameters selection, we used the following procedure: the development set or its subset was clustered many times using agglomerative clustering with all feasible hyperparameter values, and maximum BCubed-f1 value (maxB3f1) was taken as a score for the representation. This allowed us to compare clusterability of different representations while avoiding problems of selecting the number of clusters and other hyperparameters. Of course, there is a possibility that other clustering algorithms might perform better with different representations, however, we didn't see improvements from using other clustering algorithms and stick to agglomerative clustering. Table 1 shows maxB3f1 for the whole development set and for all examples of several homonyms. Evidently, dense representations are significantly better when clustering the whole development set, while sparse representations with

	dev	join@dev	build@test	follow@test	start@test
sparse	0.91	0.98	0.83	0.96	0.75
dense	0.94	0.92	0.70	0.80	0.72

Table 1: Sparse vs. dense representations, maxB3f1

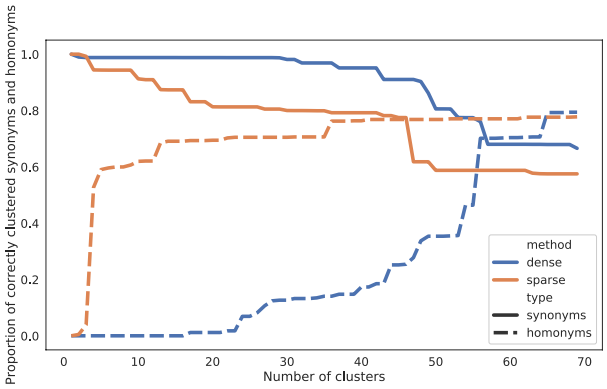

Figure 1: Recall for synonyms and homonyms w.r.t. number of clusters for dense and sparse representations

Method	#cl	PuIpuF1	B3P	B3R	B3F1
Verb baseline	227	73.94	74.61	58.95	65.86
Dense_ctx0+ss.agglo	126	76.24	60.5	**77.61**	68
Combined	239	**77.03**	**65.23**	73.82	**69.26**
@Combined+sep. *sell*	240	78.86	70.61	73.82	72.18
⋆Dense_ctx7+ss.agglo	194	77.52	66.68	**72.67**	69.55
⋆Combined2	272	**78.15**	**70.86**	70.54	**70.70**
⋆Dense_ctx7+maxsil	126	75.77	60.23	76.34	67.33

Table 2: Subtask-A, results on test. ⋆ for post-eval results, @ for manual postprocessing (out of competition)

Pattern	ctx	PuIpuF1	B3P	B3R	B3F1	maxB3F1
T and then _	2	**78.15**	**70.86**	70.54	**70.70**	**71.34**
T and _	2	77.92	70.43	70.16	70.30	71.16
_	2	77.80	70.37	69.85	70.11	71.01
T	2	77.95	68.50	**71.97**	70.19	71.15
T and then _	0	77.79	70.56	69.67	70.11	71.06
T and then _	1	77.93	70.87	69.89	70.38	**71.38**
T and then _	2	**78.15**	70.86	70.54	**70.70**	71.34
T and then _	3	78.14	70.52	**70.66**	70.59	71.29
T and then _	5	77.72	70.29	70.10	70.19	71.13
T and then _	7	77.94	**70.95**	69.89	70.41	71.24

Table 3: Subtask-A, effect of pattern and context size

an appropriate pattern are better for disambiguating homonyms.

We denote the proportion of synonyms sharing common cluster as recall for synonyms and the proportion of homonyms put in separate clusters as recall for homonyms. Figure 1 shows both metrics depending on the number of clusters for agglomerative clustering of the whole development set. It is evident that until a relatively large number of clusters (30) almost all synonyms are correctly clustered together when using dense representations, yet homonyms are clustered together as well, which gives almost 1.0 recall for synonyms and nearly 0.0 recall for homonyms. MaxB3f1 of approximately 0.94 is achieved at around 25-28 clusters (depending on the context size) where synonyms are still clustered almost perfectly. At the same time, sparse representations split homonyms into different clusters even at very small numbers of clusters, but simultaneously split synonyms also, achieving lower maxB3f1 of 0.91 in a wider range of 25-40 clusters. To solve this problem, our final solution clusters dense representations first and then splits large clusters containing examples of the same verb (to prevent splitting synonyms) into a small number of clusters to improve recall for homonyms.

Table 2 compares results on the test set. Verb baseline assigns the first token of the verb to each example as its cluster id. It overestimates the real number of clusters in the test (149), giving the highest precision but very low recall because synonyms are never clustered together. Dense representation with semi-supervised agglomerative clustering slightly underestimates the number of clusters in the test set (similarly to the development set) resulting in the highest recall due to merged synonyms. The combined approach splits some clusters hurting BCubed-recall a bit but increasing BCubed-precision, even more, resulting in better BCubed-f1. The last row shows that selecting the number of clusters which maximizes silhouette score (unsupervised approach) instead of BCubed-f1 of the labeled subset results in much worse results, hence our semi-supervised approach is beneficial. Finally, we noticed that the largest cluster had all the examples of both *sell* and *buy*, which were among the most frequent verbs in the test set. In FrameNet, they are assigned to *Commerce_sell* and *Commerce_buy* frames respectively which is a questionable solution since these are just different ways to put into words the same type of event with the same participants (something like commercial-transfer-of-property). We simply moved all examples of the verb *sell* into a separate cluster which gave significant improvement in BCubed-f1. However, this result is out of competition due to the manual postprocessing. Yet, our best result without manual postprocessing is still ranked first.

In Table 3 we report the results of clustering the test set depending on the pattern and the con-

text size used to build sparse representations at phase 2. In addition to standard metrics, we report maxB3F1 which excludes the effect of a suboptimal number of clusters selected on the comparison results. Our proposed pattern seems to give small but consistent improvement as well as context extension. The context of 1-3 sentences on both sides is a reasonable choice for sparse representations.

4 Semantic Role Induction

After looking at examples from the development set we decided that the subtask B.2 (generic semantic role induction) could be solved much more effectively using a classifier than any kind of clustering because generic roles look more like a high-level linguistic abstraction than something naturally occurring in texts. We used the development set to trained logistic regression on top of representations extracted from BERT and several hand-crafted features. BERT was pretrained in unsupervised fashion on large corpora and this results in much better generalization of our semi-supervised approach compared to a logistic regression trained only on hand-crafted features (see ablation analysis below). To select hyperparameters we used cross-validation with lexical split (i.e. there were no common verbs in train and test subsets for each fold) to ensure the best performance on new verbs not seen during training. This approach was rejected as using an additional labeled corpora to train a supervised component. However we hardly see how the development set provided by the organizers can be considered as additional.

4.1 Model Description and Results

We trained a logistic regression classifier for the 14 most frequent semantic roles in the development set. Following recommendations of Devlin et al. (2018) we used outputs from the last four layers of BERT as features. These outputs were taken for two timesteps at which the target argument and its corresponding verb were fed. To be exact, we found the first subword of the verb (for instance, *buy* for *buy out*) and the last subword for the argument (*Union* for *European Union*) performing best. Additionally we used several hand-designed features. Table 4 shows our submission results. Also, we display results when using only BERT and only hand-designed features suggesting that both of them contribute positively to the results but BERT features are much more important. For additional details regarding hand-designed features and ablation analysis please refer to Appendix B. We didn't experiment with subtask B.1 due to the lack of time, instead we used labels predicted for subtask B.2 which resulted in 64.43 / 73.11 BCubed-F1 / PuIpu-F1 compared to 45.79 / 57.99 of the best performing baseline.

Method	#cl	PuIpuF1	B3P	B3R	B3F1
ClstPerGrType	37	56.05	40.89	37.33	39.03
Logistic regression	14	**77.47**	**56.21**	**74.41**	**64.04**
w/o designed feats.	14	76.93	54.71	73.55	62.75
w/o BERT feats.	13	65.08	41.90	55.01	47.57

Table 4: Subtask-B.2, results on test

5 Conclusions

We show how neural language models can be effectively used for unsupervised inference of semantic structures. To improve the result of semantic frame induction we used a combined approach that utilizes two different vector representations, and adjusted our clustering algorithm accordingly. The design stemmed from our analysis of problems in use of neural language models for the purpose of semantic frame induction; the experiments showed that issues may be strongly related to how the models treat such linguistic phenomena as synonymy and homonymy. Designing a system that addresses this problem directly allowed us to improve the result significantly. We think that our result could be additionally improved by finding better parameters and/or model combinations. We also think that further research in this direction could lead to neural language models that explicitly address various linguistic phenomena by design, for even better inference of semantic properties.

Acknowledgments

We thank Dmitry Lipin and Dmitry Ustalov for their invaluable help, the orginizers and LDC for the inspiring task and the data, all the reviewers for the useful feedback. Alexander Panchenko has been supported by the Deutsche Forschungsgemeinschaft (DFG) within the project "Argumentation in Comparative Question Answering (ACQuA)" (grant BI 1544/7-1 and HA 5851/2-1) that is part of the Priority Program "Robust Argumentation Machines (RATIO)" (SPP-1999) and the project "JOIN-T".

References

Asaf Amrami and Yoav Goldberg. 2018. Word sense induction with neural bilm and symmetric patterns. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4860–4867. Association for Computational Linguistics.

Nikolay Arefyev, Pavel Ermolaev, and Alexander Panchenko. 2018. How much does a word weigh? Weighting word embeddings for word sense induction. In *Computational Linguistics and Intellectual Technologies. Papers from the Annual International Conference Dialogue (2018)*, pages 68–84. RSUH.

Osman Baskaya, Enis Sert, Volkan Cirik, and Deniz Yuret. 2013. Ai-ku: Using substitute vectors and co-occurrence modeling for word sense induction and disambiguation. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 300–306. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *COLING 1992 Volume 2: The 15th International Conference on Computational Linguistics*.

David Jurgens and Ioannis Klapaftis. 2013. Semeval-2013 task 13: Word sense induction for graded and non-graded senses. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 290–299. Association for Computational Linguistics.

Alexander Panchenko, Anastasiya Lopukhina, Dmitry Ustalov, Konstantin Lopukhin, Nikolay Arefyev, Alexey Leontyev, and Natalia V. Loukachevitch. 2018. Russe'2018: A shared task on word sense induction for the russian language. In *Computational Linguistics and Intellectual Technologies. Papers from the Annual International Conference Dialogue (2018)*, pages 547–564. RSUH.

Alexander Panchenko, Olga Morozova, and Hubert Naets. 2012. A semantic similarity measure based on lexico-syntactic patterns. In *KONVENS*.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, et al. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics.

Behrang QasemiZadeh, Miriam R. L. Petruck, Regina Stodden, Laura Kallmeyer, and Marie Candito. 2019. Semeval-2019 task 2: Unsupervised lexical frame induction. In *Proceedings of The 13th International Workshop on Semantic Evaluation*. Association for Computational Linguistics.

Roy Schwartz, Roi Reichart, and Ari Rappoport. 2015. Symmetric pattern based word embeddings for improved word similarity prediction. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 258–267. Association for Computational Linguistics.

Oleg Struyanskiy and Nikolay Arefyev. 2018. Neural Networks with Attention for Word Sense Induction. In *Supplementary Proceedings of the Seventh International Conference on Analysis of Images, Social Networks and Texts (AIST 2018)*, pages 208–213.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. *CoRR*, abs/1706.03762.

Dominic Widdows and Beate Dorow. 2002. A graph model for unsupervised lexical acquisition. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

build: Manufacturing/ Building	**follow**: Compliance/ Relative_time	**join**: Participation/ Becoming_a_member
drive 0.61/0.031	execute 0.53/0.0	end 0.5/0.15
export 0.5/0.031	obey 0.47/0.0	support 0.81/0.32
import 0.67/0.046	keep 0.53/0.0	continue 0.5/0.23
distribute 0.72/0.092	adopt 0.74/0.0	begin 0.44/0.21
manufacture 0.94/0.12	apply 0.68/0.013	follow 0.56/0.37
ship 0.5/0.077	maintain 0.63/0.013	lead 0.88/0.77
release 0.5/0.092	use 0.63/0.013	start 0.5/0.56
make 0.56/0.11	enforce 0.47/0.013	leave 0.5/0.82
assemble 0.78/0.18	ignore 0.42/0.013	represent 0.31/0.52
deliver 0.89/0.22	implement 0.79/0.027	rejoin 0.25/0.6
...	...	...
rebuild 0.28/0.55	confirm 0.16/0.47	buy 0.062/0.47
expand 0.33/0.75	begin 0.16/0.48	found 0.0/0.4
acquire 0.22/0.58	end 0.11/0.64	oversee 0.0/0.4
finance 0.17/0.58	see 0.053/0.43	serve 0.0/0.48
erect 0.11/0.43	include 0.053/0.55	create 0.0/0.48
open 0.11/0.77	come 0.0/0.41	acquire 0.0/0.48
fund 0.056/0.58	be 0.0/0.49	purchase 0.0/0.55
establish 0.056/0.6	, 0.0/0.51	establish 0.0/0.6
close 0.0/0.42	mark 0.0/0.53	form 0.0/0.63
start 0.0/0.43	after 0.0/0.61	become 0.0/0.82

Table 5: Examples of generated substitutes for template "T and then _"

Method	#cl	PuIpuF1	B3P	B3R	B3F1
ClstPerGrType	37	56.05	40.89	37.33	39.03
Logistic regression	14	**77.47**	**56.21**	**74.41**	**64.04**
w/o designed feats.	14	76.93	54.71	73.55	62.75
w/o BERT feats.	13	65.08	41.90	55.01	47.57
w/o BERT@arg	14	73.45	51.60	67.59	58.52
w/o BERT@verb	14	74.88	52.16	71.88	60.45
w/o verb_input_rel	14	76.92	55.45	73.45	63.19
w/o by_between	14	76.92	55.55	73.45	63.25
w/o arg_is_left	14	77.25	55.65	73.90	63.49
layer 0,1	14	73.56	50.34	68.33	57.97
layer 0	14	73.69	50.52	68.88	58.29
layer 1	14	74.71	51.65	70.47	59.61
layer 2	14	75.13	52.18	71.16	60.21
layer 4	14	76.16	54.11	72.30	61.90
layer 11	14	75.98	54.37	72.19	62.02
layer 10,11	14	76.58	55.10	73.25	62.89
layer 10	14	76.66	55.51	72.96	63.05
layer 6	14	76.98	55.72	73.29	63.31
layer 8	14	77.40	56.33	73.78	63.88

Table 6: Subtask-B.2, ablations on test set.

A Examples of generated substitutes

To show how substitutes can disambiguate homonyms we generated substitutes for examples of two most frequent frames for several verbs. For each verb we excluded rare substitutes with $P(subs|frame_i) < 0.4$ for both frames. Then we sorted the rest according to the probability ratio $\frac{P(subs|frame_1)}{P(subs|frame_2)+1e-6}$. Table 5 shows substitutes with the largest and the smallest ratio (most discriminating substitutes).

B Features and ablation analysis for Generic Semantic Role Induction subtask

We used the following hand-crafted features: an indicator that the argument is to the left of the verb and an indicator that the particle *by* is between them; categorical features for the output syntactic relation of the argument, the last relation in the path between the argument and the verb, the part of speech of the first word of the argument, the number of words and the number of words starting with a capital letter in the argument. All these features were concatenated, categorical features were encoded with one-hot vectors. In the preliminary experiments we noticed that hand-crafted features performed well by themselves but didn't improve results when concatenated with BERT outputs; this was resolved by multiplying the features by 10 (we attribute the effect to very high dimensionality of BERT outputs compared to hand-crafted features, which requires harmonizing the variance each of them adds to the scalar product in the logistic regression). We tried multiplying each feature by its own constant determined analytically from its dimensionality, but this worsened the results, so we left it for the future work.

Table 6 shows results for subtask B.2 after removing features from input representation or using different BERT layers instead of the last four. For ablation analysis, we selected L2-regularization strength using cross-validation with a lexical split after removing each feature while leaving all other hyperparameters intact. The features with largest contribution to the result are (from most to least important) BERT output at the argument, at the verb, the last relation in the path from the argument to the verb, the indicator that the particle *by* is between them (which was designed to fix errors due to passive voice) and the indicator that the argument is to the left of the verb. All other features' contributions (not shown) are small. Remarkably, removing all BERT features gives very large decrease in performance (-18 B3F1) while removing only outputs at the argument/verb gives only moderate decrease (-5.5/-3.5 B3F1) which can be explained by deeply bidirectional nature of BERT resulting in some information about both the verb and the argument present in each of these outputs. Finally, we tried using other BERT layers instead of the last four (layers 8-11) and found that intermediate layers perform best. For instance, layer 8 can replace the last four layers with very little decrease in performance, while the last two layers (10, 11) concatenated perform noticeably worse but much better than the first layers.