Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022), pages 4414–4419
Marseille, 20-25 June 2022

Are Embedding Spaces Interpretable? Results of an Intrusion Detection Evaluation on a Large French Corpus

Thibault Prouteau, Nicolas Dugué, Nathalie Camelin, Sylvain Meignier
Le Mans University, LIUM
UFR Sciences et Techniques, Le Mans Université, Avenue Laennec, 72085 LE MANS CEDEX 9, France
{first.last}@univ-lemans.fr

Abstract

Word embedding methods allow to represent words as vectors in a space that is structured using word co-occurrences so that words with close meanings are close in this space. These vectors are then provided as input to automatic systems to solve natural language processing problems. Because interpretability is a necessary condition to trusting such systems, interpretability of embedding spaces, the first link in the chain is an important issue. In this paper, we thus evaluate the interpretability of vectors extracted with two approaches: `SPINE`, a k-sparse auto-encoder, and `SINr`, a graph-based method. This evaluation is based on a *Word Intrusion Task* with human annotators. It is operated using a large French corpus, and is thus, as far as we know, the first large-scale experiment regarding word embedding interpretability on this language. Furthermore, contrary to the approaches adopted in the literature where the evaluation is performed on a small sample of frequent words, we consider a more realistic use-case where most of the vocabulary is kept for the evaluation. This allows to show how difficult this task is, even though `SPINE` and `SINr` show some promising results. In particular, `SINr` results are obtained with a very low amount of computation compared to `SPINE`, while being similarly interpretable.

Keywords: word embeddings, interpretability, word intrusion task, human evaluation, french

1. Introduction

The field of Natural Language Processing (NLP) has seen many progress with the advent of word embedding. Word embedding approaches aim to encode, syntactic and semantic information to describe the vocabulary of a corpus, in a low-dimensional space. Originally, each word was assigned a vector in this space. To train these vectors, algorithms make use of the words' co-occurrences in the corpus at hand. For instance, the `GloVe` (Pennington et al., 2014) approach is designed as a factorization of the co-occurrence matrix into two embedding matrices. More recently, contextual approaches appeared, assigning a vector to any occurrence of each word in the corpus. Among these approaches, those based on the transformer architecture have been widely adopted. We can name `BERT` (Devlin et al., 2019) for English, and `CamemBERT` (Martin et al., 2020) for French. These approaches allowed great improvements of the performances of NLP systems on downstream tasks such as Named Entity Recognition or POS tagging. However, the architectures of these systems are far greater than those developed ten years before, and thus harder to interpret, even if Clark et al. (2019) used probing to successfully uncover the operation of attention heads.

This paper focuses on the interpretability of word embeddings. Interpretability is a major issue for artificial intelligence systems. In particular, we consider that interpretability is a necessary condition to trusting these systems. Because embeddings are often used as input for NLP systems, we consider these vectors as the first building block of these architectures. We thus emphasize the importance of their interpretability to build interpretable systems from start to finish. Instead of considering post-hoc or probing systems to explain the embeddings, we focus on systems that are interpretable by design (Rudin, 2019). Furthermore, we aim to evaluate the interpretability of two recent systems, `SPINE` and `SINr` described in Section 2, in an actual experimental setup. Based on a French news corpus with a large vocabulary, we propose to quantitatively evaluate the interpretability of these two systems. To that aim, we consider a human evaluation using the intrusion detection task as recommended in the literature detailed Section 2. As far as we know, this is the first experiment on interpretability of such systems on the French language, and with such a large vocabulary.

In Section 2, we briefly review the literature regarding interpretable word embedding approaches. We also describe how the word intrusion task is used to evaluate interpretability of these approaches. Then in Section 3, we describe our experimental setup, in particular we describe the corpus at hand, and the models with their parameters. Section 4 presents the results. We observe that this task is particularly difficult, and that, in this setup with a huge vocabulary, they are lower than in previous work by Subramanian et al. (2018). Besides, we further discuss these results, emphasizing on the encouraging results of `SINr` (Prouteau et al., 2021), especially when considering its very low run time. Finally, we conclude by describing the perspectives of this work.

2. Related Work

Interpretability of Word Embeddings. In 2012, before the advent of models such as `Word2Vec` or `GloVe` to extract efficient word embeddings, Murphy et al. (2012) already introduced interpretable embeddings. The model they introduced, `NNSE` (Non negative sparse embedding) is quite close to `GloVe`: factorizing the co-occurrence matrix, decomposing it into the product of two matrices, $\mathcal{A}$ the embedding matrix and $\mathcal{D}$ the new basis. However, the matrix is embedded in a 1000-dimension space, and in the manner of a sparse coding, they introduce a l_1 regularization on the embedding matrix to enforce sparsity of the embeddings. This seminal paper introduced the key ideas:

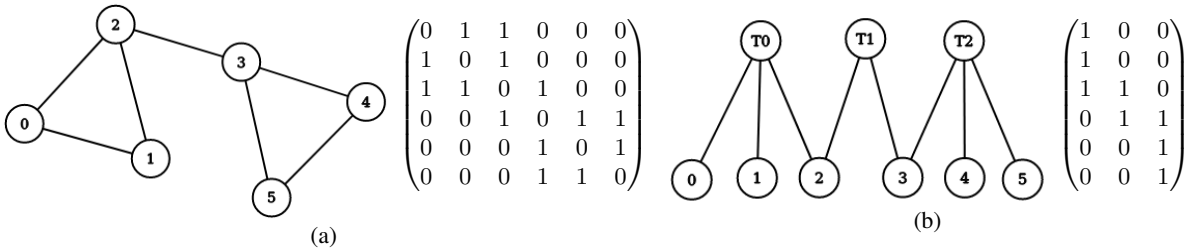

Figure 1: Illustrating the LDBGF: (a) a toy graph and its adjacency matrix (b) a low-dimensional bipartite graph representation. The adjacency matrix of the $\perp$ nodes (rows) from the toy graphs are represented with their links to the $\top$ nodes (columns). The resulting adjacency matrix is smaller and is actually an interpretable embedding: dimensions represent the connectivity to $\top$ nodes which are tangible entities.

representing the data in a high-dimensional space (more than 300), enforcing sparsity and non-negativity. In such a model, each dimension is supposed to encode a specific theme, and thus words are defined only using a small set of these themes (values set to 0 for the remaining dimensions). In 2015, Faruqui et al. (2015) introduced SPOWV, that benefits both from these ideas and of the progress made in the field with Word2Vec and GloVe. Indeed, SPOWV stands for Sparse Overcomplete Word Vectors representations: authors detailed an overcomplete sparse coding of dense pre-trained word embeddings. In other words, from Word2Vec or GloVe vectors, they extract sparse representations enforcing them using regularization terms. These vectors are so-called overcomplete, because the output space is bigger (from 500 to 3000) than the input one (300). Then, in 2018, Subramanian et al. (2018) introduced SPINE (Sparse Interpretable Neural Embeddings). In this paper, such as with SPOWV, authors sparse code pre-trained word embeddings. However, the methodology is quite different, SPINE relies on overcomplete (hidden layer bigger than input and output) auto-encoders, introducing sparsity losses to enforce the interpretability.

Finally, the authors of this paper introduced SINr (Prouteau et al., 2021), a graph-based approach to compute graph and word embeddings. They designed LDBGF (Lower-dimensional Bipartite Graph Framework) described in Figure 1: based on a graph representation of the co-occurrence matrix, one aims to compute the smallest bipartite version of this graph, namely with the fewest top nodes (significantly less than nodes in the original graph). The SINr approach is a fast approximation of this method that relies on community detection (graph clustering, Blondel et al. (2008)) leading to sparse word representations. Contrary to SPINE and SPOWV, SINr does not rely on pre-trained representations, in that sense, it is more similar to NNSE.

Word Intrusion Task. Most word embedding models are deemed efficient on intrinsic evaluation tasks such as the well-known similarity task, analogy task or based on results obtained on downstream tasks (*e.g.* document classification, sentiment analysis *etc*). Although these tasks are commonly used to infer the efficiency of a model, they do not give insight regarding the interpretability of their dimensions. The task commonly called *Word Intrusion Task*, first introduced by Chang et al. (2009), was created specifically to assess this aspect in the framework of topic modeling. The method was later transposed to the field of word embeddings (Murphy et al., 2012; Faruqui et al., 2015; Luo et al., 2015; Subramanian et al., 2018; Bourgeade et al., 2021). The goal of the *Word Intrusion Task* for word embeddings is to evaluate the semantic coherence of the dimensions in the embedding space. A coordinate is a column in the embedding matrix—related to a dimension of the embedding space. A dimension is said semantically coherent if a human can find the odd one (intruder) out of a set of words with the highest values on a coordinate. A coordinate is selected at random and the words are sorted by decreasing order of the value they have for this coordinate. Ideally, in an interpretable dimension, the top words should be thematically related, we select them as data for a task. An intruder is selected at random according to the following conditions : the intruder must be in the 10% of top words in coordinate of another dimension and the bottom 50% in the coordinate of the dimension evaluated. All the words selected (top words and the intruder) are then shuffled and presented to a human who must select an intruder. The accuracy of the detection of the *intruders* on multiple coordinates sheds light on whether the dimensions of a model are semantically coherent and subsequently interpretable. Table 1 presents some tasks extracted from each model (Section 3) used in the evaluation.

Model	Top Words			Intruder
SPINE	suffrage	urne *(ballot box)*	législative *(legislative)*	colmatage *(sealing)*
	tramway	ferroviaire *(rail)*	rail	orientation
SINr-1	monospace *(multipurpose vehicle)*	véhicule *(vehicle)*	voiture *(car)*	remarquer *(to notice)*
	droit *(law)*	mort *(death)*	peine *(penalty)*	fortune
SINr-2	réseau *(network)*	chaîne *(channel)*	groupe *(group)*	déclencher *(trigger)*
	Intel	microprocesseur *(microprocessor)*	processeur *(processor)*	garder *(to keep)*

Table 1: Examples of tasks extracted for each model.

So far, the *Word Intrusion Task* was mostly applied on sparsified word embeddings extracted from English corpora and with a limited vocabulary of 3,000 and 15,000 words for SPINE (Subramanian et al., 2018) and SPOWV (Faruqui et al., 2015), 40,000 words for NNSE (Murphy et al., 2012).

3. Experimental Setup

Intrusion Detection. In our evaluation, due to the sparsity of the SINr model presented in the next section and the unprecedented size of vocabulary for such an evaluation, we lowered the proportion of top words to three (as opposed to 4 or 5) and the intruder is sampled in the bottom 30% of values in the coordinate. Because we are working with a large vocabulary spanning decades, we decided to exclude the numerous named entities except for the organizations that can easily be interpreted. For SINr, due to the high dimensionality of the vectors, the probability of selecting a dimension is positively correlated with the number of words that have a non-null value for this dimension. For SPINE, the probability of selection follows a uniform law. In total 200 dimensions were sampled for each of the three systems evaluated. Once the dimensions are selected, we extracted the top 3 words on each corresponding coordinate along with an intruder, following the requirements defined earlier.

For each word, three possibilities are presented to the evaluator $(-, \pm, +)$, the goal is to detect the intruder. If the annotator can easily find out the intruder from the list of four words, the "+" choice should be selected. When the annotator is not sure about the intruder, but seems pretty sure about what words are not intruders, "−" should be selected for the words that are less likely to be intruders. Finally, "±" is added to indicate that the word is probably the intruder, but, differs from "+", the annotator indicates that some hesitations remain. This scale allows a fine-grained analysis of the interpretability (hesitations between words, seemingly coherent dimension despite the intruder *etc.*) for each dimension rather than just binary—whether the intruder was found or not.

The *Word Intrusion Task* was served through a custom web interface built using the open-source package Label-Studio (Tkachenko et al., 2020 2021). Figure 2 shows an example of a task in the developed interface.

Evaluation Participants. The models were evaluated by 19 individuals in their 20s completing a master's degree in NLP. The participants have prior knowledge of word embedding methods and their evaluation. All are literate in French and have lived for at least four months in France—we did not observe a difference in the results between French natives and non-natives. They were each presented with 99 tasks to complete (33 per model evaluated) within a set duration of one hour. The order of the tasks and words in each annotation track are randomized for each participant, the evaluators ignore which model they are evaluating with each task. Each task was solved by three or four evaluators, in total, 1881 evaluations have been collected.

The homogeneity in the profiles of participants is beneficial in the sense that they are all in the same age group and have a similar linguistic experience, they also grasp the concepts (semantic similarity and proximity) related to the evaluation of such systems. Furthermore, the number of participants may help to alleviate some caveats of similar studies, namely relying on a limited crowd of annotators that have helped to develop the system or (paid) crowdsourcing.

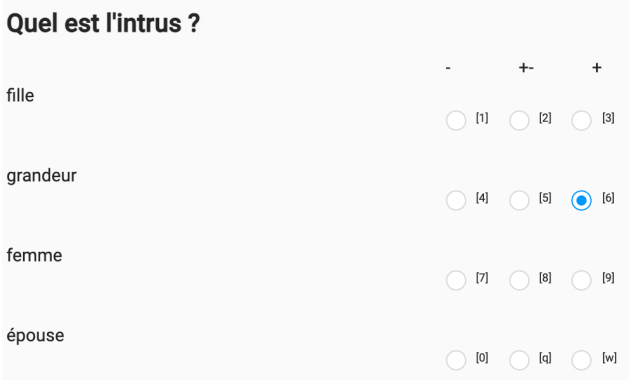

Figure 2: Example of an intrusion task as presented to the annotators. Translation of words ordered as presented: girl, size, woman, wife.

Data. Corpora is paramount to compute word embeddings. We wish to proceed to an evaluation with a larger vocabulary than what was previously done in the literature, we used a collection of news articles from the newspaper *Le Monde* (Monde, 2005), the news agency *AFP* and news articles crawled from French news websites (*WEB*). The articles from *Le Monde* span from 1987 to 2006, the articles from *AFP* are dated from 1994 to 2006, the articles for *WEB* were crawled in 2007.

The corpus was preprocessed using the fr_core_news_lg model available with *SpaCy* (Honnibal et al., 2020). The text is lemmatized, named entities are detected and grouped under a single token if they contain multiple words, stop words are removed along with words occurring fewer than 10 times. After preprocessing, the training corpus contains 330M tokens for a vocabulary of 323k words.

Models. In this paper, we focus on two distinct approaches. The first one is SPINE described in Section 2, it is based on a sparse autoencoder that aims to sparse code pre-trained vectors. This approach was proved to obtain better interpretability results than SPOWV and NNSE when considering the intrusion task, while being competitive when used in downstream tasks. We thus use it as a solid baseline for our evaluation. However, it was previously evaluated on an English corpus, with a much smaller vocabulary (15k words). In this paper, we consider French, and a more realistic vocabulary size to be able to represent the wide diversity of the vocabulary. SPINE is controlled by a few hyper-parameters, such as the sparsity fraction, the hidden dimension size, the λ scalars that controls regularization, and the number of epochs. We used an implementation (Danovitch, 2020) running on GPU. We set these parameters to the values used in the implementation and obtained by Subramanian et al. (2018) using a grid search: λ are set to 1, the dimension H to 1000 and the sparsity fraction to 0.15, the number of epochs to 4000. Subramanian et al. (2018) showed that SPINE performs better when Word2Vec vectors are provided as inputs, we thus pre-train Word2Vec vectors on our corpus (window size of 5, 300 dimensions and 5 epochs).

The second approach we evaluate is SINr (Prouteau,

2020–2021). In this graph-based approach, words are represented by nodes, and their co-occurrences by weighted links. This method then relies on community detection to cluster nodes that are more connected together than with the rest of the graph. Vectors are then extracted by exploiting for each node, its connectivity to all the clusters. The size of the vectors thus depends on the clustering. The clustering approach used is the hierarchical Louvain (Blondel et al., 2008) that greedily optimizes the modularity, a quality metric. The only parameter of the approach is the level of the hierarchy used to extract the vectors. In Prouteau et al. (2021), the SINr space is intrinsically evaluated using level zero—the lower one, with most communities. In this evaluation, we consider the first level with 22807 dimensions that we name SINr-1, and the second one with 4708 dimensions that we name SINr-2.

4. Results

In our setup, human annotators have more than the choices *intruder* and *not an intruder*. They can also express their hesitation between two words, or express the fact that it seems there is no intruder. This makes the analysis of the results a bit more difficult, but in our humble opinion, it allows to get relevant data to assess the interpretability of the models considered. In Table 2, we take into account the specificity of our evaluation and present three rows: the first one is the percentage of tasks when the annotator found the intruder (IntruderOK), the second one is the percentage of tasks when the annotator found the intruder or has hesitated between the intruder and another word (HesitateOK), the third one (Consistent) adds to the previous figure the tasks where the annotator considered that all the words were in the same lexical field. Results are detailed for SPINE, SINr-1 (22,8k dimensions) and SINr-2 (4,7k dimension). As one can see, results for IntruderOK are quite low, about one third of the tasks are correctly annotated without any hesitations. Subramanian et al. (2018) obtained 75% of accuracy in their paper introducing SPINE. However, their evaluation was binary, and most importantly, was made on a selected vocabulary of 15k most frequent words. This evaluation is thus more realistic with 323k words of vocabulary, it shows how difficult this task is in this realistic framework. When adding cases were annotators hesitated between two words, with one being the intruder, more than half of the tasks are resolved correctly by annotators. Results show that the performances of SINr are on par with those of SPINE: SPINE does perform better considering only the IntruderOK accuracy, but SINr has better results when considering IntruderOK + HesitateOK + Consistent case. In particular, it seems that there are more Consistent cases with SINr, probably due to its number of dimensions which is substantially higher, especially in the SINr-1 model. In this model, there may be, for instance, more redundancy between dimensions. Furthermore, there may be fewer words expressing themselves on the dimensions, the intruder picked randomly being closer to the top 10%

In Table 3, we further analyze the results presenting the cases where annotators were not able to find the intruder.

	SPINE	SINr-1	SINr-2
IntruderOK	**36%**	31%	35%
+ HesitateOK	56%	53%	**60%**
+ Consistent	57%	**58%**	**62%**

Table 2: Positive results of the intrusion detection task.

The IntruderKO case is when annotators were certain of having found the intruder but were wrong, HesitateKO when they were hesitating between two words that were both of them not the intruder, and No consistency when they were not seeing any consistency between any of the four words presented.

Similarly to the positive results, the negative results show that there are no large discrepancies between SPINE and SINr. When considering the IntruderKO, SINr-2 has the lowest number of intruders that were wrongly chosen, SPINE is in between SINr-1 and SINr-2. Regarding HesitateKO, the proportions for all the models are similar. No consistency results show that SINr has fewer tasks where annotators did not find any semantic coherence between words than SPINE. Because the results are close for SINr-1 and SINr-2, we can hypothesize that the dimensionality of vectors doesn't have a significant impact on the number of No consistency tasks.

	SPINE	SINr-1	SINr-2
IntruderKO	14%	18%	**12%**
HesitateKO	**10%**	**10%**	11%
No consistency	19%	**14%**	15%

Table 3: Negative results of the intrusion detection task.

The inter-annotator agreement is presented in Table 4. We can see that overall, in 56% of tasks, at least two annotators agree on the same annotation—same box ticked for the same task. At the model level, SPINE has the highest agreement for a pair of annotators and three annotators with respectively 58 and 21 percent. The two SINr models obtain the same result (55%) when considering pairs of annotators. SINr-2 is behind for three annotator agreement with 13% despite better results on the task, this lower agreement may be related to the higher proportion of hesitations (36%) than for SINr-1 and SPINE (respectively 32% and 30%) resulting in more diverse annotations. We also computed *Fleiss' kappa* (Fleiss, 1971): the overall task κ is 0.23. For the three models evaluated, the κ is of the same magnitude: 0.26 for SPINE, 0.24 for SINr-1 and 0.21 for SINr-2. The κ scores obtained are related to a *fair agreement* (Landis and Koch, 1977), considering the freedom willingly left to the annotators—three choices for each word in each task—reaching a high agreement is unlikely. Moreover, the *Word Intrusion Task* in its original setting is already difficult, subsequently, allowing more choices for the annotators and evaluating larger models adds another layer of complexity. Despite the added complexity, the agreement scores obtained show that in more than 50% of tasks; at least two annotators reach a similar decision.

To conclude about the task's difficulty, Figure 3 presents the response time according to the rank of the task—ordered as

Overall	SPINE	SINr-1	SINr-2
56%, 17%	**58%, 21%**	55%, 17%	55%, 13%

Table 4: Inter-annotator agreements across all models presented and overall for the *Word Intrusion Task*. For each model, the first value is the percentage of tasks where at least two evaluators annotated similarly. The second value is the percentage of tasks where the three evaluators annotated similarly.

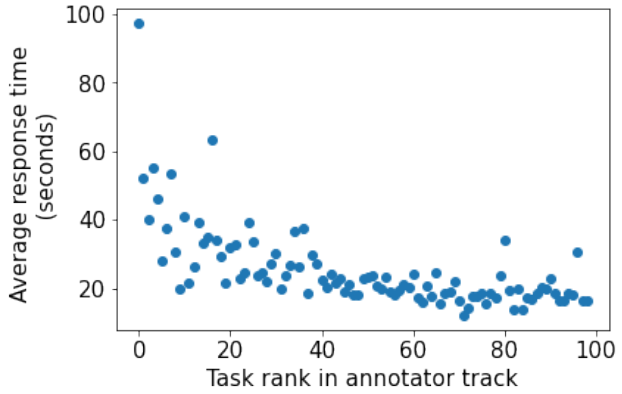

Figure 3: Average response time over all the annotators depending on the rank of the task solved in annotator track.

they were presented to each annotator. We can see that there seems to be a learning curve to the *Word Intrusion Task*. At the beginning, the participants take on average, more time to submit a response than at the end of the evaluation. This shows that participants gain confidence as the evaluation progresses.

In summary, SPINE and both SINr models have similar results, we can see that SINr can compete on the grounds of interpretability. Although these results are encouraging, there is still a long way to go for the interpretability of word embeddings, especially when adopting a framework closer to reality—with a large vocabulary.

Discussion. The results obtained for the *Word Intrusion Task* show that a graph framework like SINr can compete with complex neural network methods while maintaining a significantly shorter run time. Table 5 shows the run time in hours for each model, the run times were computed on an Intel Xeon Gold 6226R 2.90GHz CPU, allowing 20 CPU threads for all the jobs and 120GB of RAM, SPINE was trained on the same server with the same settings and a Nvidia Quadro RTX 6000 GPU. The results show that SINr runs roughly 17 times faster than SPINE. There is a difference in run time between SINr-1 and SINr-2 due to the number of clusters and consequently larger dimension of vectors to be computed. The run time is a partial indicator of the power consumption to train word embeddings, SINr runs mostly on a single CPU core to train word embeddings in a short time, whereas SPINE requires a GPU.

On the other hand, there is a trade-off between interpretability and efficiency on the similarity task for SINr. Due to the unavailability of similarity datasets in French, we can only evaluate word embeddings in English, and hence

Model	Runtime
Word2Vec + SPINE	17.2
SINr-1	1.3
SINr-2	**1**

Table 5: Runtime of each model in hours.

use British National Corpus (BNC Consortium, 2007)—thereafter BNC. We ran SINr on the BNC, a well-known collection of documents in English which totals 100M tokens. Prior to learning the word embeddings, the text was lemmatized and words occurring fewer than 5 times were discarded. Based on a similarity evaluation on classical datasets: RG65 (Rubenstein and Goodenough, 1965), MEN (Bruni et al., 2014), MTurk (Halawi et al., 2012) and WS-353 (Finkelstein et al., 2001) for different levels of the hierarchical Louvain algorithm, we can see Table 6 that the efficiency on the similarity task seems correlated with the number of communities. This correlation may be the consequence of the reduction in the number of communities leading to bigger communities that are less specific. On the other hand, when considering interpretability, we observe Section 4 that SINr-2 with a lower number of dimensions seems to produce more interpretable dimensions than SINr-1. The challenge for SINr is to gain insight into where the cursor on vector dimensionality shall be set: trading interpretability off for performance or vice versa.

Number of communities	RG65	MEN	MTurk	WS-353
67,260	0.73	0.63	0.53	0.53
18,867	0.61	0.56	0.48	0.48
4,674	0.48	0.51	0.45	0.44

Table 6: Similarity evaluation of SINr word embeddings for different partition sizes of the hierarchical Louvain (Section 3) trained on the British National Corpus (BNC). Results are Pearson correlations (higher is better).

5. Conclusion

We conducted an evaluation that is, to our knowledge, the most extensive on the interpretability of dimensions of word embeddings, in terms of vocabulary size, and also the first evaluation of this type on the French language. The goal was to assess the interpretability of models resulting from SPINE, a solid baseline, and SINr, a graph-based embedding method. On the same principle as the original *Word Intrusion Task*, the annotators are asked to find an intruder among words that should be semantically related, however, compared to the original setting, they have the choice to state a hesitation between two words or exclude words they are confident are not the intruder. This is an added benefit as it allows a fine-grained analysis of the interpretability of the dimensions of a model. The results of this evaluation show that despite the complexity of the task at hand, SPINE and SINr produce some interpretable dimensions, even when trained on a large French

corpus. Our analysis shows that SINr competes with SPINE, while maintaining a low computational time and thus a more moderate use of resources. However, there is still a long way to go before attaining the interpretability of word embedding models, some challenges remain related to the performances on more classic evaluation tasks and the use of these models as the first brick of machine learning algorithms. For instance, with SINr, there seems to be a trade-off between interpretability and performance on the similarity task. Such tasks require evaluation datasets that are costly to construct and consequently unavailable for most languages outside of English. To overcome these challenges, it would be interesting to come up with automated evaluation methods not based on datasets or humans that are intrinsically language-independent. Especially for word intrusion, Lau et al. (2014) laid the first stone towards an automatic evaluation of the interpretability of word embeddings.

Acknowledgement

This work was partly funded by ANR-19-CE38-0012 GEM and ANR-21-CE23-0010 DIGING.

6. Bibliographical References

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008.

Bourgeade, T., Muller, P., and Van de Cruys, T. (2021). Plongements Interprétables pour la Détection de Biais Cachés. In *TALN*, pages 64–80.

Chang, J., Gerrish, S., Wang, C., Boyd-graber, J., and Blei, D. (2009). Reading Tea Leaves: How Humans Interpret Topic Models. In *Advances in Neural Information Processing Systems*, volume 22.

Clark, K., Khandelwal, U., Levy, O., and Manning, C. D. (2019). What does bert look at? an analysis of bert's attention. *arXiv preprint arXiv:1906.04341*.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]*, May. arXiv: 1810.04805.

Faruqui, M., Tsvetkov, Y., Yogatama, D., Dyer, C., and Smith, N. A. (2015). Sparse Overcomplete Word Vector Representations. In *ACL*, pages 1491–1500.

Fleiss, J. L. (1971). Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382.

Landis, J. R. and Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, 33 1:159–74.

Lau, J. H., Newman, D., and Baldwin, T. (2014). Machine Reading Tea Leaves: Automatically Evaluating Topic Coherence and Topic Model Quality. In *EACL*, pages 530–539.

Luo, H., Liu, Z., Luan, H., and Sun, M. (2015). Online Learning of Interpretable Word Embeddings. In *EMNLP*, pages 1687–1692.

Martin, L., Muller, B., Ortiz Suárez, P. J., Dupont, Y., Romary, L., de la Clergerie, E., Seddah, D., and Sagot, B. (2020). CamemBERT: a Tasty French Language Model. In *ACL*, pages 7203–7219.

Murphy, B., Talukdar, P., and Mitchell, T. (2012). Learning Effective and Interpretable Semantic Models using Non-Negative Sparse Embedding. In *COLING*, pages 1933–1950.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543.

Prouteau, T., Connes, V., Dugué, N., Perez, A., Lamirel, J.-C., Camelin, N., and Meignier, S. (2021). SINr: Fast computing of Sparse Interpretable Node representations is not a sin! In *IDA*, number 12695, pages 325–337.

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215.

Subramanian, A., Pruthi, D., Jhamtani, H., Berg-Kirkpatrick, T., and Hovy, E. (2018). SPINE: SParse Interpretable Neural Embeddings. In *AAAI*.

7. Language Resource References

BNC Consortium. (2007). *British National Corpus, XML edition*.

Bruni, E., Tran, N. K., and Baroni, M. (2014). Multimodal Distributional Semantics. *Journal of Artificial Intelligence Research*, 49:1–47, January.

Danovitch, J. (2020). SPINE - Sparse Interpretable Neural Embeddings. Open source implementation available from https://github.com/jacobdanovitch/SPINE.

Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., and Ruppin, E. (2001). Placing search in context: The concept revisited. In *Proceedings of the 10th International Conference on World Wide Web*, page 406–414.

Halawi, G., Dror, G., Gabrilovich, E., and Koren, Y. (2012). Large-scale learning of word relatedness with constraints. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 1406–1414.

Honnibal, M., Montani, I., Van Landeghem, S., and Boyd, A. (2020). spaCy: Industrial-strength Natural Language Processing in Python.

Le Monde. (2005). *Text corpus of "Le Monde"*. distributed via ELRA: ELRA-Id W0015, ISLRN 421-401-527-366-2.

Prouteau, T. (2020–2021). SINr - Sparse Interpretable Node representations. Open source implementation available from https://git-lium.univ-lemans.fr/tprouteau/sinr.

Rubenstein, H. and Goodenough, J. B. (1965). Contextual correlates of synonymy. *Commun. ACM*, 8(10):627–633.

Tkachenko, M., Malyuk, M., Shevchenko, N., Holmanyuk, A., and Liubimov, N. (2020-2021). Label Studio: Data labeling software. Open source software available from https://github.com/heartexlabs/label-studio.