Cross-Lingual Idiom Sense Clustering in German and English

Mohammed Shayaan Absar
absarshayaan@gmail.com

Abstract

Idioms are expressions with non-literal and non-compositional meanings. For this reason, they pose a unique challenge for various NLP tasks including Machine Translation and Sentiment Analysis. In this paper, we propose an approach to clustering idioms in different languages by their sense. We leverage pre-trained cross-lingual transformer models and fine-tune them to produce cross-lingual vector representations of idioms according to their sense.

1 Introduction

Idiom handling is an important aspect of any NLP system due to the unique way idioms can affect the meaning of a sentence. Due to their non-compositional meanings, NLP systems need to treat idioms as a single lexical unit. In Machine Translation, in particular, current transformer models tend to struggle when translating an idiom because of this. Experiments (Dankers et al., 2022) show that transformers (Vaswani et al., 2017) often fail to treat idioms in this manner and instead translate them compositionally resulting in poor translations.

This paper approaches the problem of clustering idioms in different languages based on their sense. Through this, we aim to improve semantic representations of idiomatic expressions to aid NLP tasks that rely on accurate sense disambiguation. In this paper, we make use of pre-trained cross-lingual language models (Conneau et al., 2020) to do this.

Our approach involves fine-tuning these models to generate cross-lingual vector representations of idioms based on sense. These representations can then be used to form sense clusters of idioms.

This idea can be further extended by leveraging idiom databases e.g. (Villavicencio et al., 2004) to identify the sense of an idiom not present in the database. By finding an idiom within the same cluster that is already in the database, we can infer the sense of the unknown idiom.

Idioms that share the same sense share a common meaning beyond their literal interpretations. Machine Translation systems often treat idioms compositionally and produce translations that are too literal and don't make sense in the translated text. The absence of parallel idiom datasets often hinders the effective training of transformers to address this challenge. We feel that our approach could aid this. Instead of training models to translate idioms in isolation which is often not practical, we propose a method capable of grouping idioms by their shared meaning. This enables the models to understand the meanings these idioms convey and the relationships between them across languages.

To evaluate our approach we conduct experiments using multi-lingual idiom datasets and assess the results.

2 Approach

In this paper, we employ BERT (Devlin et al., 2019) models that are pre-trained specifically for cross-lingual contexts to facilitate our approach. We fine-tune the model by training it on a dataset consisting of English idioms and corresponding German idioms. We developed a dataset of roughly 14,000 English and German Idioms for this purpose.

In order to train the model, we load the dataset and create *translation clusters* which consist of idioms that are direct translations of one another (taken from the website `dict.cc`). During training, we try to ensure the sense vectors of idioms in the same translation cluster are close to one another in order to create effective sense clusters.

We make use of the XLM-RoBERTa (Conneau et al., 2020) model which is trained on 2.5 terabytes of data in 100 different languages. XLM-RoBERTa

Proceedings of the First NLP4TIA Workshop, pages 13–19
Varna, Sep 8, 2023
https://doi.org/10.26615/978-954-452-091-5_003

Idiom	Gloss
es mit Fassung tragen	(to bear it with composure)
take it on the chin	
grin and bear it	
gute Miene zum bösen Spiel machen	(make a good face for the bad game)
in den sauren Apfel beißen	(bite into the sour apple)

Table 1: An example of a translation cluster. The gloss is provided for reference and is not part of the dataset.

Type	Count
English	6912
German	7763
Total	14675

Table 2: Dataset Statistics.

is trained with the multilingual MLM (Masked Language Model) objective. This allows the model to understand bi-directional context within text. This bi-directional context understanding is particularly crucial when dealing with idioms. XLM-RoBERTa produces contextual representations of the tokens that are passed to it. We then utilise pooling and an additional linear layer, to generate vector representations of the idioms.

We then employ a variety of clustering techniques to form sense-based clusters

2.1 Dataset

We hand-collected the dataset from the website `dict.cc`. We made use of a 90-5-5 train-test-validation split. Table 2 shows the composition of the dataset.

2.2 Model Architecture

The model architecture (Figure 1) consists of the XLM-RoBERTa model followed by a pooling layer and a linear layer which generates the phrase level embeddings. We made use of batch normalization (Ioffe and Szegedy, 2015) and weight decay (Loshchilov and Hutter, 2019) to make training more stable and reduce overfitting.

We investigate the effects of different pooling methods.

3 Training

3.1 Fine-Tuning

In order to train the model, we fine-tune the XLM-RoBERTa model and learn the weights for the final linear layer. We make use of the Adam optimizer (Kingma and Ba, 2017) during this process.

3.2 Triplet Loss

The triplet loss (Schroff et al., 2015) is defined as:

$$\mathcal{L} = \max(0, dist(a, p) - dist(a, n) + \alpha)$$

where:

- a is the anchor sample.
- p is a positive sample (same translation cluster as anchor).
- n is a negative sample (different translation cluster from anchor).
- $dist$ is the distance metric between samples.
- α is the margin that controls the minimum desired separation.

Triplet loss solely considers the distance between the anchor, positive and negative vectors. Some loss functions for the task of learning embeddings also consider the angle between the vectors (Wang et al., 2017). However, we felt that triplet loss worked well enough for our task.

4 Training Experiments

4.1 Embedding Dimensions

We investigated the effect of the number of nodes in the final linear layer (the number of dimensions of the sense embeddings that are produced) on training. As seen in Figure 2, the training is fairly similar for all of the embedding dimensions that we tested with the 64 and 128 dimensions performing the best. However, upon examining the validation losses, we found that the models with smaller embedding dimensions performed poorly. In our final model, we used an embedding size of 64.

4.2 Activation Functions

We also investigated the effects of different activation functions on the final linear layer. As seen in Figure 3, ELU, ReLU, Leaky ReLU and sigmoid

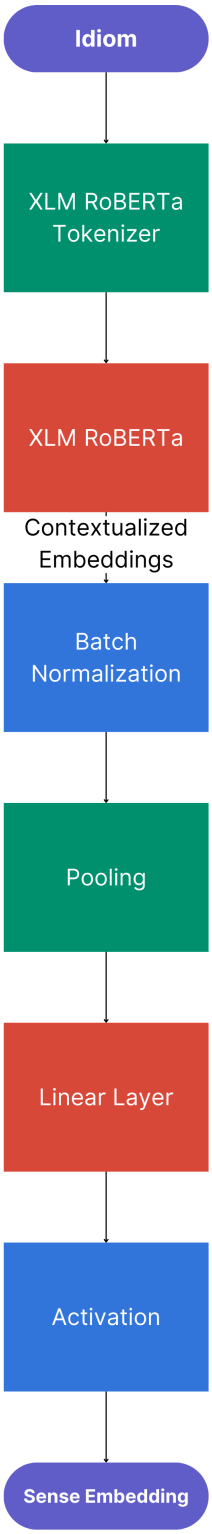

Figure 1: Model Architecture

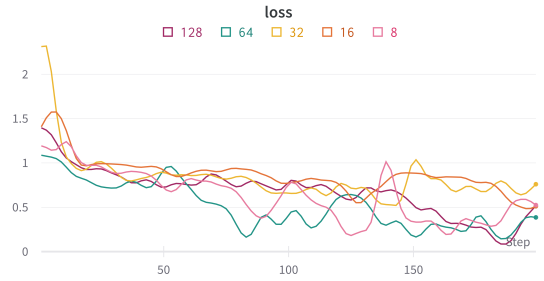

Figure 2: Loss (triplet) during training with different embedding dimensions.

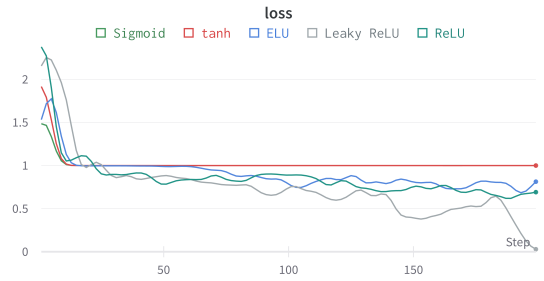

Figure 3: Loss (triplet) during training with different activation functions in the last layer.

all perform reasonably well with tanh performing poorly.

The reason for the poor performance could be due to the tendency of tanh to saturate hindering training.

We decided to use the Leaky ReLU activation function for our final model as it produced the most consistent results during the training process.

4.3 Learning Rate

After investigating the effect of the learning rate (Figure 4) on the training process, we found that a lower learning rate led to improved performance and convergence. For our final model, we used a learning rate of 0.00001.

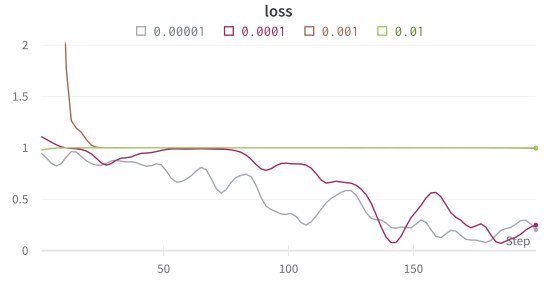

Figure 4: Loss (triplet) during training with different learning rates.

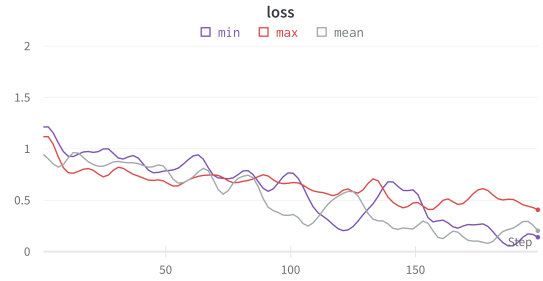

Figure 5: Loss (triplet) during training with different pooling methods.

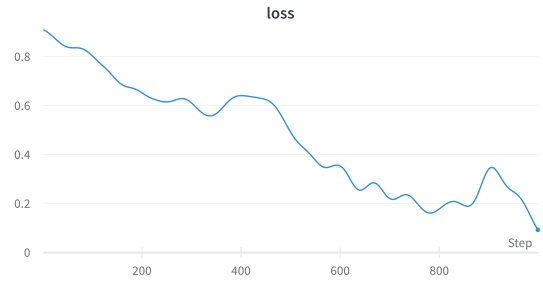

Figure 6: Loss during the training of the final model.

4.4 Pooling Method

Our investigations into the effects of different pooling methods on training (Figure 5) show that minimum pooling leads to the smallest loss. However, the validation losses for minimum pooling were inconsistent and mean pooling performed much better. For our final model, we used mean pooling.

4.5 Final Model Hyperparameters and Design Choices

Table 3 shows the design choices and hyperparameters of our final model. Figure 6 shows loss during the training of our final model.

Hyperparameter	Value
Batch Size	64
Weight Decay Rate	0.1
Learning Rate	0.00001
Embedding Dimensions	64
Linear Layer Activation	Leaky ReLU
Pooling Method	Mean
Training Epochs	1000

Table 3: Design choices and hyperparameters of our final model.

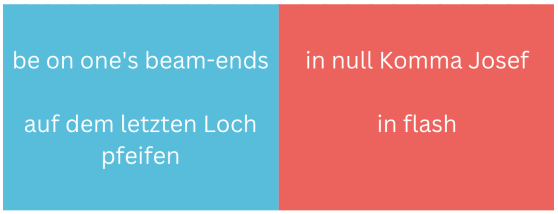

Figure 7: An extract from our clustering tests showing performance on clustering direct translations.

Figure 8: An extract from our clustering tests showing performance on clustering idioms of similar sense.

5 Clustering

We made use of the test data and applied various clustering algorithms to the encodings produced by the model. We made use of the K-means clustering, DBSCAN (Ester et al., 1996) and Bisecting K-Means (Steinbach et al., 2000) algorithms.

5.1 Direct Translations

We found that the model performed very well when attempting to cluster idioms that are direct translations of one another.

As seen in Figure 7, the model is able to effectively cluster idioms that are direct translations of one another.

5.2 Sense Clustering

Although the model is generally able to detect idioms with similar senses, it does struggle in some cases.

As seen in Figure 8, the model sometimes fails to properly cluster idioms of similar sense. 'kreuzfidel' (meaning to be as happy as a king) and 'feel like a kid in the candy store' both suggest a positive feeling and 'put one's nose in other people's business' and 'auf die Nüsse gehen' (meaning to get on someone's nerves) both have a negative sense. However, in this case, they were placed in different clusters.

We felt the failure was due to the choice of loss function. By using positive and negative samples, there is only a binary relationship between idioms.

This means the model fails to capture the nuanced similarities and differences between the idioms.

We also believe that the model weights relationships between idioms in the same language too heavily, which may hinder its ability to effectively cluster the idioms by their sense. This bias can result in clusters heavily dominated by a single language.

6 Model Evaluation

6.1 UMAP Projections

We utilised UMAP (McInnes et al., 2020) to project a subset of the sense vectors into 2 dimensions. This dimensionality reduction enables us to see more clearly the relationships the model is (and isn't) capturing.

Figure 9 shows the UMAP projection of the embeddings produced by the fine-tuned model and Figure 10 shows the UMAP projection of XLM-RoBERTa before the fine-tuning process. The idioms with the same colour in the graph are translations of one another so should be close together (if the model was trained effectively). The figures show that idioms that are translations of one another appear significantly closer to one another in the fine-tuned model. This indicates the model is capable of learning the semantic similarities between idioms in different languages as a result of the fine-tuning process.

6.2 Mean Reciprocal Rank

To assess the performance of the final model, we employed the Mean Reciprocal Rank (MRR) metric. We treated the sense embedding of a given idiom as a query and the sense embedding of the translation of that idiom as a target. We calculated MRR values on both the fine-tuned model and XLM-RoBERTa before fine-tuning so we could examine the effects of fine-tuning. By applying this technique, we aimed to gauge the model's effectiveness in placing idioms close to translations of themselves in a vector space. The results are shown in the Table 4.

From the data provided in the table, it's evident that the fine-tuning process had a significant positive impact on the model's performance. The MRR values for the fine-tuned model consistently outperformed those of the model without fine-tuning. This suggests that the fine-tuning process effectively enhanced the model's ability to generate sense-based vector representations of idioms.

Test No.	Batch Size	MRR before fine-tuning	MRR after fine-tuning
1	26	0.2184	0.4771
2	36	0.1205	0.3138
3	40	0.0698	0.1825

Table 4: The results of our MRR tests.

7 Conclusion and Future Work

In conclusion, our study presented a method of clustering idioms in different languages by their sense, making use of pre-trained transformer models. Our experiments show that our model works effectively but struggles in some circumstances.

During our tests, we found the model sometimes failed to cluster idioms of similar sense together. This can be partly attributed to the binary nature of triplet loss which fails to capture degrees of similarity between idioms.

Additionally, we identified a potential bias in the model's weighting of relationships between idioms in the same language.

To address these issues, further work can be done to mitigate these issues. We will work towards developing better loss functions and finding methods of reducing the bias.

Limitations

While our model shows promise at cross-lingual idiom sense clustering, we feel that there is room for improvement. This can partly be improved by larger datasets. By incorporating more diverse and comprehensive idiomatic expressions from different languages, the model can learn more robust representations and better capture the nuances of idiomatic senses.

Additionally, we believe that a more sophisticated loss function could further enhance the model's clustering capabilities. Instead of considering binary relationships between idioms, this loss function would consider the degree of relatedness between the idioms. This would allow the model to consider varying degrees of similarity between idioms resulting in better performance.

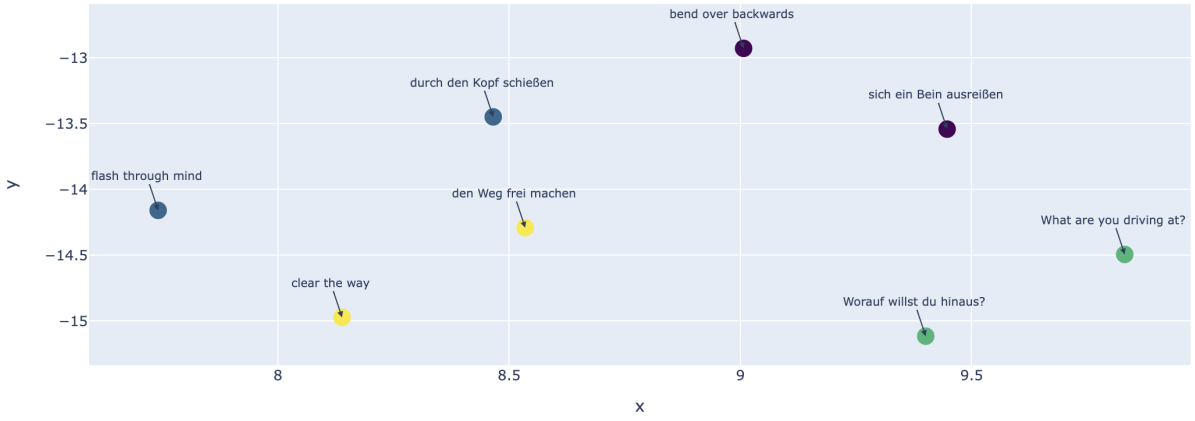

Figure 9: UMAP Projection of sense embeddings produced by the fine-tuned model.

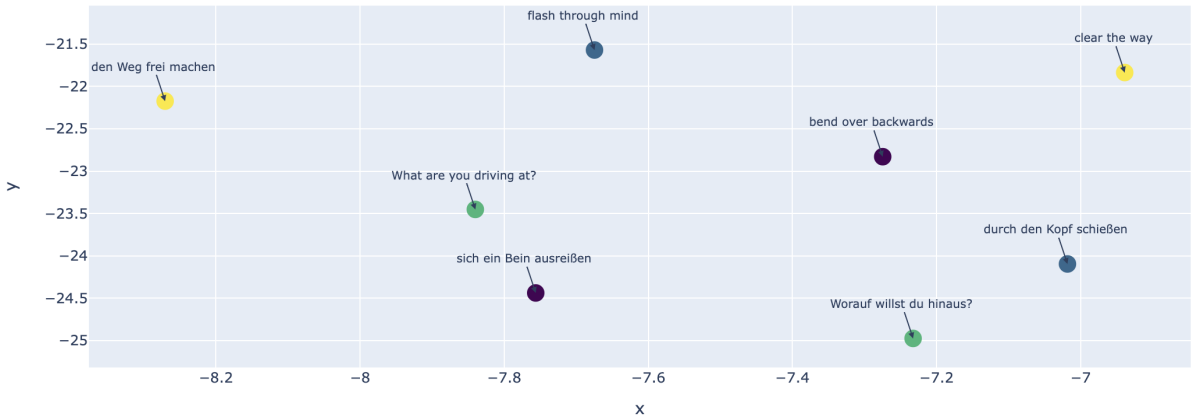

Figure 10: UMAP Projection of sense embeddings produced by XLM-RoBERTa before fine-tuning.

References

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Verna Dankers, Christopher Lucas, and Ivan Titov. 2022. Can transformer be too compositional? analysing idiom processing in neural machine translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3608–3626, Dublin, Ireland. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization.

Leland McInnes, John Healy, and James Melville. 2020. Umap: Uniform manifold approximation and projection for dimension reduction.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. IEEE. [link].

Michael Steinbach, George Karypis, and Vipin Kumar. 2000. A comparison of document clustering techniques.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Aline Villavicencio, Timothy Baldwin, and Benjamin Waldron. 2004. A multilingual database of idioms. In *LREC*.

Jian Wang, Feng Zhou, Shilei Wen, Xiao Liu, and Yuanqing Lin. 2017. Deep metric learning with angular loss.