Multilingual Embedding Alignment

Introduction

- Embeddings are the basic ingredient in many kinds of natural language processing tasks.
- When it comes to multilingual tasks, one of the challenges is that the embedding spaces are not aligned even if the spaces have been shown to have a similar geometric arrangements as Mikolov et al. show in [2], specially when the training process is similar.
- The alignment is required for two kinds of embedding models.
  - One is the embedding models separately trained on monolingual data and,
  - The other type is multilingual models trained on parallel multilingual data.
- As far as the multilingual models are concerned, most of the times the training process itself implicitly encourages for the alignment. [6, 10]
- On the other hand, the monolingual models are concerned, the alignment has to be done explicitly after the models are trained.
- Even though multilingual embedding models are getting popular nowadays, the monolingual embedding alignment is still vital specially when it comes to,
  - Low resource languages,
  - When pre-training or fine tuning a multilingual model is time and resource consuming and also,
  - When well trained monolingual models are already available.

Types of Embeddings

- Word Embeddings - EX: Word2Vec, Glove, FastText
- Sentence Embeddings - Ex: Doc2Vec, S-BERT, Universal Sentence Encoder
- Knowledge Graph Embeddings

Types of Alignments

- Word-to-Word
- Sentence-to-Sentence
- Knowledge-to-Knowledge

Word-to-Word Alignment Techniques

- Supervised Methods
  - Regression Models (Ex: least square minimization [2])
  - Orthogonal Models [3]
  - Margin Models (Ex: contrastive loss [10], triplet loss [9])
- Unsupervised Methods
  - Adversarial learning methods [5, 11]
- Semi-Supervised Methods

Exploiting Similarities among Languages for Machine Translation [2]

- A paper in 2013 by Mikolov et al. - 1571 citations
- They have show that the embedding spaces of two independently trained languages using word2vec have similar geometric arrangements.
- And therefore through a simple linear transformation, the two embedding spaces can be aligned together.

$$\min_{W} \sum_{i=1}^{n} \|W x_i - z_i\|^2$$

- Where, ***W*** is the transformation matrix to be found and ***(xi, zi)*** are dictionary words of the source and target languages.
- Authors have used the top most frequent words of the source language to find ***W***.
- Authors have used WMT11 dataset which is a translation dataset to prove their work.

Results (check the paper)

- See the paper
- Accuracy of word translation compared to other traditional techniques. (Table 2)
- Translation accuracy of infrequent words (Figure 4)
- Accuracy with the cosine distance threshold (Table 3, 4)

Normalized Word Embedding and Orthogonal Transform for Bilingual Word Translation [3]

- A paper by Xing et al. in 2015 - 420 citations
- The authors have enhanced the accuracy of the same experiment done in the previous paper by applying embedding vector normalization and by changing the linear transformation optimization to an orthogonal transformation.
- The arguments were,
    - Most of the time embedding comparisons are done using cosine distance.
    - Original word2vec models were not trained to minimize optimize cosine distances but to optimize the inner product. (since the output embeddings are not normalized)
    - Therefore the training objective and the inference are **not consistent**.
- Therefore they have trained the word2vec with with normalized vectors and have done the same optimization criteria.
- Therefore the linear transformation becomes an orthogonal transformation when doing the alignment in order for ***zi*** to be a normalized vector.

$$\max_{W} \sum_{i} (W x_i)^T z_i .$$

Results (check the paper)

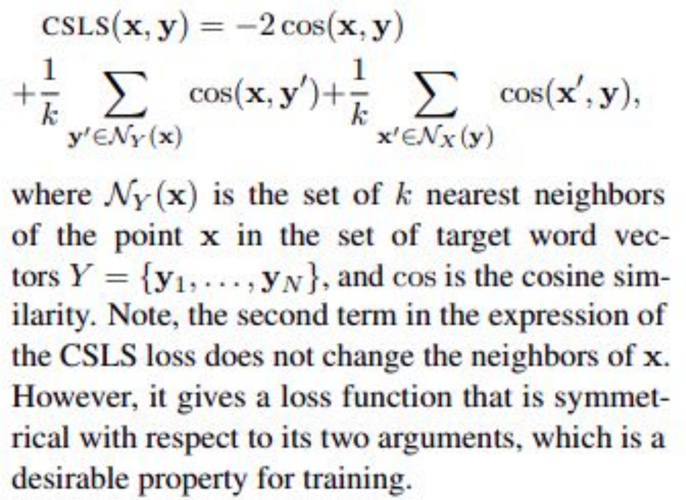
- See the paper
- Accuracies compared with the previous paper (Table 1, 2)

Loss in Translation: Learning Bilingual Word Mapping with a Retrieval Criterion [4]

- A paper by Mikolov et al. in 2018 - 256 citations
- Authors have addressed the so called “hubness problem” in embedding alignment.
- Hubs are words that appear too frequently in the neighborhoods of other words.
- There have been solutions to mitigate this issue at inference time by using a different criteria (loss) such as Inverted Softmax (IFS) or Cross domain similarity local scaling (CSLS), rather than using the same criteria used at training phase.
- Using a different criteria at inference adds an inconsistency.
- Therefore the authors have included the CSLS criteria directly to the training objective.

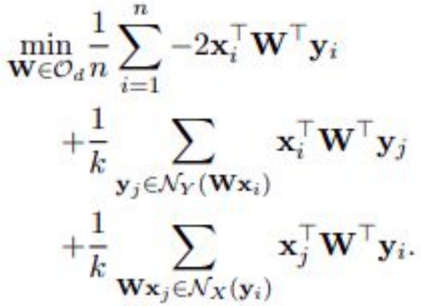
Loss in Translation (contd.)

- They have used the vector normalization and orthogonal transformation matrix same as in the previous paper.
- The major difference is, instead of simple inner product optimization, they have used CSLS loss (Bottom left figure).
- When the normalized orthogonal transformation is assumed, this optimization objective looks like in the bottom right criteria where ***Od*** is the manifold of orthogonal matrices.
- The authors have shown that this objective can further be relaxed to be an convex optimization problem.
- This is the algorithm fast text has used in their embedding alignments.



$$\mathrm{CSLS}(\mathbf{x},\mathbf{y}) = -2\cos(\mathbf{x},\mathbf{y}) + \frac{1}{k}\sum_{\mathbf{y}'\in\mathcal{N}_Y(\mathbf{x})}\cos(\mathbf{x},\mathbf{y}') + \frac{1}{k}\sum_{\mathbf{x}'\in\mathcal{N}_X(\mathbf{y})}\cos(\mathbf{x}',\mathbf{y}),$$

where $\mathcal{N}_Y(\mathbf{x})$ is the set of k nearest neighbors of the point $\mathbf{x}$ in the set of target word vectors $Y = \{\mathbf{y}_1, \ldots, \mathbf{y}_N\}$, and cos is the cosine similarity. Note, the second term in the expression of the CSLS loss does not change the neighbors of $\mathbf{x}$. However, it gives a loss function that is symmetrical with respect to its two arguments, which is a desirable property for training.



$$\min_{\mathbf{W}\in\mathcal{O}_d}\frac{1}{n}\sum_{i=1}^{n} -2\mathbf{x}_i^\top\mathbf{W}^\top\mathbf{y}_i + \frac{1}{k}\sum_{\mathbf{y}_j\in\mathcal{N}_Y(\mathbf{W}\mathbf{x}_i)}\mathbf{x}_i^\top\mathbf{W}^\top\mathbf{y}_j + \frac{1}{k}\sum_{\mathbf{W}\mathbf{x}_j\in\mathcal{N}_X(\mathbf{y}_i)}\mathbf{x}_j^\top\mathbf{W}^\top\mathbf{y}_i.$$

Results (check the paper)

- See the paper

References

[1] T. Mikolov, K. Chen, G. Corrado, J. Dean, ‘Efficient estimation of word representations in vector space’, arXiv preprint arXiv:1301. 3781, 2013.

[2] T. Mikolov, Q. V. Le, I. Sutskever, ‘Exploiting similarities among languages for machine translation’, arXiv preprint arXiv:1309. 4168, 2013.

[3] C. Xing, D. Wang, C. Liu, Y. Lin, ‘Normalized word embedding and orthogonal transform for bilingual word translation’, στο Proceedings of the 2015 conference of the North American chapter of the association for computational linguistics: human language technologies, 2015, pp. 1006–1011.

[4] A. Joulin, P. Bojanowski, T. Mikolov, H. Jégou, E. Grave, ‘Loss in translation: Learning bilingual word mapping with a retrieval criterion’, arXiv preprint arXiv:1804. 07745, 2018.

[5] T. Schuster, O. Ram, R. Barzilay, A. Globerson, ‘Cross-lingual alignment of contextual word embeddings, with applications to zero-shot dependency parsing’, arXiv preprint arXiv:1902. 09492, 2019.

[6] G. Lample A. Conneau, ‘Cross-lingual language model pretraining’, arXiv preprint arXiv:1901. 07291, 2019.

[7] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, ‘Bert: Pre-training of deep bidirectional transformers for language understanding’, arXiv preprint arXiv:1810. 04805, 2018.

[8] A. Vaswani et al., ‘Attention is all you need’, Advances in neural information processing systems, τ. 30, 2017.

[9] N. Reimers I. Gurevych, ‘Sentence-bert: Sentence embeddings using siamese bert-networks’, arXiv preprint arXiv:1908. 10084, 2019.

[10] F. Feng, Y. Yang, D. Cer, N. Arivazhagan, W. Wang, ‘Language-agnostic bert sentence embedding’, arXiv preprint arXiv:2007. 01852, 2020.

[11] A. Conneau, G. Lample, M. Ranzato, L. Denoyer, H. Jégou, ‘Word translation without parallel data’, arXiv preprint arXiv:1710. 04087, 2017.