Instructions for EMNLP 2020 Proceedings

Anonymous EMNLP submission

Abstract

The analogy task introduced by Mikolov et al. (2013) has become the standard method to evaluate word embeddings because it correlates well with performance on downstream tasks an The analogy task is not suitable for low-resource languages, however, because good performance requires a large amount of training data and meaningful analogies can be difficult to create. As a result, researchers in working in low-resource settings have created many ad-hoc methods for

As a result, those working in low resource or specialized settings have lacked a standard method to evaluate the quality of word embeddings. To address this problem, we propose two different intrinsic evaluation methods, which achieve greater performance sensitivity than the analogy task in low resource settings. We demonstrate the flexibility of these methods by using them to tune embeddings for 18 low resource languages as well as the full list of unicode emojis. Furthermore, our methods are designed such that custom test sets can be developed semi-automatically in over 400 different languages, which helps increase access for research in areas previously inhibited by low amounts of data.

1 Introduction

Word vector embeddings are widely used in Natural Language Processing (NLP), and have become a critical tool for achieving state-of-the-art performance in many tasks ().

Mikolov et al. (2013) showed that the word2vec model produces embeddings with useful linear structure, and this structure can be used to solve analogy tasks.

$$\texttt{king} - \texttt{man} + \texttt{woman} \approx \texttt{queen}. \tag{1}$$

To evaluate the quality of their word embeddings, Mikolov et al. (2013) created a 14 category evaluation set commonly called the Google Analogy Test Set. Importantly, good performance on this analogy test set correlates with good performance on downstream tasks like sentiment analysis (), part of speech tagging (), and named entity recognition (). Improvements to the standard word2vec algorithm such as FastText (Bojanowski et al., 2016) and GloVe (Pennington et al., 2014) have demonstrated their efficacy by demonstrating improvements on the Google Analogy Test Set.

At the same time, only small consideration has been given to areas where the analogy task either cannot be applied or fails to capture meaningful differences in quality between sets of embeddings.

Bias-variance tradeoff in dimensionality of word embeddings (Yin and Shen, 2018)

1.1 Weaknesses of Google Analogy Test Set

People have talked about the weaknesses of the google analogy set, ie unbalanced between semantic and syntactic categories, mostly syntactic, when we usually care more about semantics. ()

1.2 Low-resource word embeddings

The analogy task is not suitable for low-resource languages, however, because good performance requires a large amount of training data and meaningful analogies can be difficult to create. As a result, researchers in working in low-resource settings have created many ad-hoc methods for

English can be a low resource setting when studied diachronically (e.g. Hamilton et al., 2016b,a; Kutuzov et al., 2018; Kozlowski et al., 2019; Dubossarsky et al., 2017; Tang, 2018; Szymanski, 2017; Liang et al., 2018; Chen et al., 2017).

Azarbonyad et al. (2017) considers splits over different political ideologies.

(Kulkarni et al., 2015) similar. Along this line of work, Gonen et al. (2020) is the most similar to ours. They propose a new method for comparing the similarity of two word embeddings in order to find words that are used differently. Their method assumes that quality word embeddings have already been trained, and our evaluation metrics provide a way to ensure that this training is done well.

Mnih and Kavukcuoglu (2013) improves the word2vec skipgram model with noise contrastive estimation (NCE). They report requiring 10x less computation and 4x less training data to get the same results. Gupta et al. (2019) introduce an extension to the skipgram and fasttext models that uses kernel PCA to reduce the amount of data needed during the training procedure. (i.e., their method requires less data to train good models.) They still use the

Due to Zipf's law, some words will be very infrequent. Analogies are known to now work well on these infrequently used words.

1.3 Highly multilingual word embeddings

Al-Rfou et al. (2013) trained word2vec embeddings on 100 different languages using Wikipedia as the training data. Grave et al. (2018) extended this work by training FastText embeddings on 157 languages using data from the Common Crawl project.

1. For both papers, the Hindi language had the smallest amount of training data, with 23 million and 1.8 billion tokens, respectively. In Section **??**, we consider languages with significantly less training data. For example, the Ancient Hindi language contains only 0.6 million tokens, and we are still able to generate meaningful word embeddings using our evaluation metrics.

2. Due to the difficulty of evaluating so many languages, however, the authors evaluate the embeddings only on 10 languages, and the other 147 remain unevaluated.

Most work regarding word embeddings has been carried out in resource rich settings. In particular, the original word2vec embeddings were trained on a GoogleNews corpus containing 6 billion English tokens of which 783 million were unique (Mikolov et al., 2013). Unsurprisingly, greater amounts of training data leads to more accurate embeddings. But in reality, there are many domains in which this amount of data is impractical or even impossible to obtain. Consider for example, a low resource language such as Telugu or a historical language like Latin. Furthermore, resource rich languages can quickly become low resource when considered

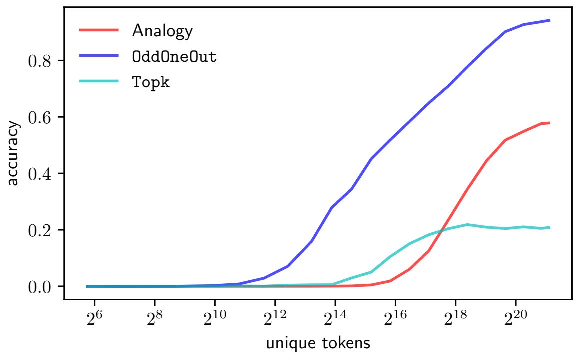

Figure 1: The plot above compares the regions of effectiveness for our evaluation metrics. The analogy task fails to measure change in accuracy of the embeddings until the number of unique words in the training dataset reaches 2^{16}, much later than both `OddOneOut` and `Topk`. Though `OddOneOut` seems the clear victor of these methods, experimentation in 3 shows that `Topk` works better in some circumstances.

from a specialized perspective. Those studying the evolution of language over time often require very specific corpora which in turn restricts the quantity of data.

More recently, embeddings have even been generated for tokens that serve the purpose of words, but are not considered words in the traditional sense. The clearest example of these non-traditional embeddings is emoji, which have emerged as an important feature of natural language data in social media. In this setting it is unclear whether the analogy task should be applied and if so how to do so systematically and at scale.

In this paper, we demonstrate that the analogy task is not always suitable for evaluating quality of word embeddings, particularly when training resources are limited and when working in niche or specialized domains. As a solution to this problem, we propose `OddOneOut` and `Topk`, two alternative evaluation methods which provide greater usability in low resource settings and are more general in nature. To further increase flexibility in deployment of embedding evaluation systems we also present Wikidata as a tool for generating test sets for these methods in up to 461 languages.

The remainder of the paper is organized as follows. Section 2 introduces the `Topk` and `OddOneOut` tasks. Section 3 Section 4

2 Evaluation Methods

From a technical standpoint, our methods are quite simple and designed to be as generally applicable as possible. Both `OddOneOut` and `Topk` require a test set which consists of groups of categories where each category contains words that are semantically similar. The `OddOneOut` method seeks to identify the word that does not belong when presented with a group of words. For a given word, `Topk` finds the k most similar words according to the model and compares them to words that belong to the semantic category from which the word was chosen.

We now formalize the methods. Assume that there are m categories, that each category has n words, that there are v total words in the vocabulary, and that the words are embedded into $\mathbb{R}^d$. Let $c_{i,j}$ be the jth word in category i, and let $C_i = \{c_{i,1}, c_{i,2}, ..., c_{i,n}\}$ be the ith category.

2.1 The Topk method

Let $\texttt{FindMostSimilar}(k, w)$ return the k most similar words in the vocabulary to w. We define the `Topk` score for class i to be

$$\texttt{Topk}(k,i) = \frac{1}{n}\sum_{j=1}^{n}\frac{1}{k}\sum_{x\in\texttt{FindMostSimilar}(k,c_{ij})}\mathbb{1}[x \in C_i] \quad (2)$$

and the `Topk` score for the entire evaluation set to be

$$\texttt{Topk}(k) = \frac{1}{m}\sum_{i=1}^{m}\texttt{Topk}(k,i). \quad (3)$$

The runtime of `FindMostSimilar` is $O(dvk)$.[1] So the runtime of $\texttt{Topk}(k,i)$ is $O(dnk^2v)$ and the runtime of $\texttt{Topk}(k)$ is $O(dnmk^2v)$. Typically k is small (we recommend $k = 3$ in our experiments), and so the runtime is linear in all of the interesting parameters. In particular, it is linear in both the size of our input data and evaluation set.

2.2 The OddOneOut method

Define the `OddOneOut` score of a set S with k words with respect to a word $w \notin S$ as

$$\texttt{OddOneOut}(S,w) = \mathbb{1}[w = \hat{w}], \quad (4)$$

where

$$\hat{w} = \arg\max_{x\in S\cup\{w\}} \|x - \mu\| \quad (5)$$

$$\text{and} \quad \mu = \frac{1}{k+1}\left(w + \sum_{i=1}^{k} s_i\right). \quad (6)$$

We define the kth order `OddOneOut` score of a category i to be

$$\texttt{OddOneOut}(k,i) = \frac{1}{|P|}\sum_{(S,w)\in P}\texttt{OddOneOut}(S,w) \quad (7)$$

where

$$P = \{(S,w) : S \text{ is a combination of } k \text{ words from } C_i, \text{and } w \in V \quad (8)$$

In Equation (8) above, the total number of values that S can take is $\binom{n}{k} = O(n^k)$, and the total number of values that w can take is $O(v)$, so $|P| = O(n^k v)$. Finally, we define the k-th order `OddOneOut` score of the entire evaluation set to be

$$\texttt{OddOneOut}(k) = \frac{1}{m}\sum_{i=1}^{m}\texttt{OddOneOut}(k,i). \quad (9)$$

The runtime of $\texttt{OddOneOut}(S,w)$ is $O(dk)$. So the runtime of $\texttt{OddOneOut}(k,i)$ is $O(dkn^k v)$ and the runtime of $\texttt{OddOneOut}(k)$ is $O(dkmn^k v)$.

Comparing the runtimes of $\texttt{OddOneOut}(k)$ and $\texttt{Topk}(k)$, we can see that $\texttt{OddOneOut}(k)$ is a factor of $O(m^{k-1}k^{-1})$ slower. In real world applications, $m \gg k$, and so `OddOneOut` will take considerably more time to compute. In particular, the Mikolov test evaluations in Section 3 below use $m = 50$ and $k = 3$, so the $\texttt{OddOneOut}(k)$ score takes approximately 800x longer to compute.

3 Experiments

We demonstrate the usefulness of our evaluation metrics with two experiments. First, we show that the `OddOneOut` and `Topk` metrics are better measures of word embedding quality than the analogy metric in the low-resource regime. Second, we show that the `OddOneOut` and `Topk` metrics are useful for model selection in an emoji embedding task where the analogy task cannot be applied. This experiment also demonstrates that the `OddOneOut` and `Topk` metrics correlate with downstream task performance.

3.1 Small dataset word embeddings

This experiment measures the performance of the `OddOneOut`, `Topk`, and `Analogy` metrics as

[1] We use gensim's implementation of `FindMostSimilar`, which uses the naive loop strategy for computing the nearest neighbor. Data structures like the kd-tree or cover tree could potentially be used to speed up this search, but we did not find such data structures necessary.

a function of data set size. ***FIXME:*** *Figure 1 shows that the* `OddOneOut` *and* `Topk` *metrics are* `Analogy`*.*

For training data, we use a 2017 dump of the English-language Wikipedia that contains 2 billion total tokens and 2 million unique tokens. The dataset is freely distributed with the popular gensim library (Řehůřek and Sojka, 2010) for training word embeddings, and it is therefore widely used. State-of-the-art embeddings are trained on significantly larger datasets—for example, datasets based on the common crawl contain hundreds of billions of tokens even for non-English languages (Buck et al., 2014; Grave et al., 2018)—but since our emphasis is on the low-resource setting, this 2 billion token dataset is sufficient.

Using the wikipedia dataset, we generate a series of synthetic low-resource datasets of varying size. First, we sort the articles in the wikipedia dataset randomly.[2] Then, each dataset i contains the first 2^i tokens in the randomly ordered wikipedia dump.

On each of these low-resource datasets, we train a word2vec skipgram model with gensim's default hyperparameters[3], which are known to work well in many contexts. Importantly, we do not tune these hyperparameters for each low-resource dataset. Instead, we use the same hyperparameters because our goal is to isolate the effects of dataset size on the three evaluation metrics.

FIXME: *In order to apply the evaluation metrics, we need test sets for each metric. We cannot use the exact same test set because the* `Analogy` *metric requires a test set with different formatting than the* `OddOneOut` *and* `Topk` *metrics. In the* `Analogy` *task, we are given a set of tuples of the form*

$$cat1a, cat1b, cat2a, cat2b$$

where the

To ensure a fair comparison, we use the Google Analogy Test Set on the `Analogy` *directly, and construct a modified version of this dataset that can be used for the* `OddOneOut` *and* `Topk` *metrics.*

We use the Google Analogy Test set. The format of the test sets for the `Analogy` *task is different than the format for the* `Topk` *and* `OddOneOut` *tasks. The* `Analogy` *task test set is forma We speculate that because the analogy task is measuring this extra information, it requires more training data.*

3.2 Emoji Experiments

This second experiment demonstrates the versatility of our methods by applying them to the domain of emoji embeddings.

Emoji embeddings are an important topic of study because they are used to improve the performance of sentiment analysis systems (Felbo et al., 2017; Barbieri et al., 2017; Ai et al., 2017; Wijeratne et al., 2017; Al-Halah et al., 2019; Eisner et al., 2016, e.g.). Unfortunately, the standard analogy task is not suitable for evaluating the quality of emoji embeddings for two reasons. First, emoji embeddings are inherently low-resource—only 3000 unique emojis exist in the Unicode standard—and thus evaluation techniques specifically designed for the low-resource setting will be more effective. Second, the semantics of most emojis do not allow them to be used in any analogy task. In particular, the original emoji2vec paper (Eisner et al., 2016) identifies only ***FIXME:*** *5* possible emoji analogies. In order to tune their emoji embeddings, Eisner et al. (2016) therefore do not use the analogy task, and instead use an ad-hoc "emoji-description classification" method that required the creation of a test set with manually labeled emotion-description pairs.

Recent work Guntuku et al. (2019) combines uses the subdivision technique (Section **??**) to study how emoji are used differently in different parts of the world.

The simple and general design of our methods allows them to be easily adapted to intrinsically evaluate emoji embeddings. We demonstrate this by training our own emoji embeddings using the code from Eisner et al. (2016), but we use `OddOneOut` and `Topk` as the model selection metric. The semantic categories of emojis are trivial to generate from unicode's full emoji list.[4]

In addition to intrinsic evaluation of emoji2vec with their custom method, (Eisner et al., 2016) also deploy their vectors in a downstream sentiment analysis of tweets. We verify the performance of our tuned emoji embeddings by also evaluating

[2] This sorting is required for our low-resource datasets to be representative of English language text. Without this random sorting step, most of our datasets would be based only on articles that begin with the letter `A`, and therefore would not contain a representative sample of English words.

[3] Embedding dimension 100, number of epochs 1, learning rate 0.025, window size is 5, min count is 5. ***FIXME:*** *should we defined min count?*

[4] https://unicode.org/Public/emoji/13.0/emoji-test.txt

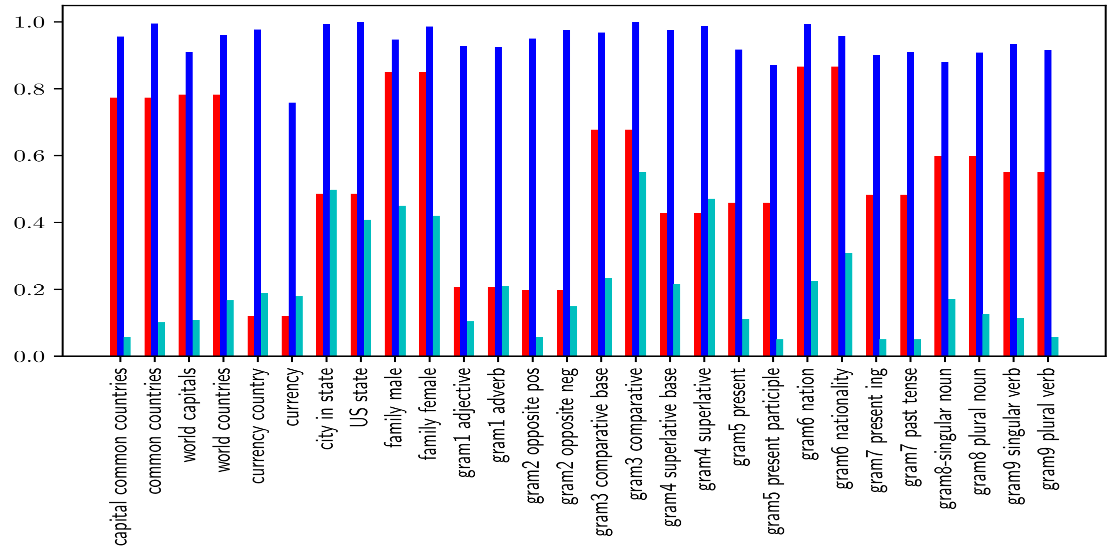

Figure 2: The methods perform better on some categories than others. `Topk` seems to excel in categories that are more homogenous like 'family female', while analogies seem to work best with geographical relationships. Note that in adapting the Google analogy set to work with our methods required splitting each relationship pair into two separate categories. As a result the analogy score for a given relationship is shown twice; one bar in each of the categories that make up the pair.

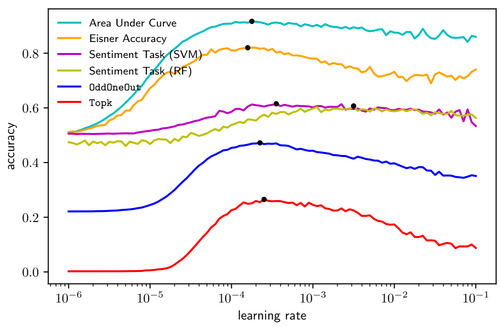

Figure 3: This figures shows the tuning of emoji embeddings across different learning learning rates where the max accuracy for each metric is marked with a point. Both `Topk` and `OddOneOut` follow the shape of (Eisner et al., 2016) area under the curve and accuracy metrics. Our methods lead us to select essentially the same hyperparameters as (Eisner et al., 2016) and reproduced results on the downstream task.

performance on this same sentiment analysis task. Figure 3 shows the results of evaluation, we find that our embeddings were able to reproduce (Eisner et al., 2016) performance on the downstream task.

4 Multilingual Content Analysis

Third, we use the `OddOneOut` and `Topk` metrics to select the hyperparameters for models trained on 17 ancient languages provided by the CLTK () library. This experiment

In Section 4 below, we use these

4.1 Automatic Test Set Generation with Wikidata

One of the most difficult and time consuming steps in the process of generating of high quality word embeddings is the creation of a comprehensive test set. Consequently, one of the biggest advantages of the `OddOneOut` and `Topk` methods is that compatible test sets can be semi-automatically generated in hundreds of languages. This is possible using Wikidata. This publicly available SPARQL database contains a comprehensive structure of the semantic content contained in Wikipedia along with its relationship to other items. Simple queries constructed using the Wikidata Query Service can be used to return semantic categories of words that can be included in a test set.

Using Wikidata we were able to reconstruct all of the semantic categories contained within the Google analogy set with the option of greater customization and more categories. Additionally, Wikidata supports the translation of queried items into 461 languages, allowing test sets to easily be converted between languages. Though not all items have translated labels in all 461 languages, support is quite extensive and will only get better.

One of the disadvantages of using Wikidata is that syntactic categories are much harder to construct; however, since semantic categories are usually more difficult to generate and require more arbitrary decisions this approach is still very valuable.

4.2 Classical Languages Experiments

An obvious application of `OddOneOut` and `Topk` is in training models for under-resourced languages. The Classical Language Toolkit (CLTK) (et al., 2014–2019) is one such ongoing project supporting nlp for various classical, low resource, and often historical languages. To confirm the efficacy of our methods in the low resource setting, we train and tune word embeddings on all languages supported by the Classical Language Toolkit which contain at least one downloadable corpus and a tokenization tool. In total this leaves us with 18 low resource languages.

For each language we preprocess all corpora available through CLTK, using their normalization and tokenization tools where appropriate. We then perform a randomized grid search over the key model parameters consisting of 100 samples. The key hyperparameters we searched over can be found in Table 1. After training the model we perform an intrinsic evaluation for the embeddings on the corresponding language specific test set and pick the model that had the highest `Combined Score` which is calculated as the harmonic mean of `OddOneOut` and `Topk`.

Although the parameter combinations searched over for each language were the same, the best model for each language contained a unique set of parameters. This supports the need for an intrinsic tuning method for word embeddings so that highest performance can be achieved. The tuning results in Table 1 also seem to indicate that fastText models perform better than word2vec in low resource scenarios due to the training of n-grams.

Aside from intrinsic evaluation, our methods can be leveraged as a qualitative downstream task we call multilingual content analysis. The goal of the task is to determine the degree to which some concept or category is captured by a set of word embeddings. We implement this by using Wikidata to construct 3 categories of interest–Biblical Figures, Facets of Hinduism, and Facets of Buddhism–and then evaluate them using `OddOneOut` and `Topk`. The performance on this task demonstrates the degree to which our models understand the semantics of the chosen topics and provides insight regarding the content they were trained on.

With intentionally selected categories we are able draw out the differences in content between the different language models that in many cases matches our intuition. By evaluating on a category of biblical figures we see highest performance from the Hebrew and Latin models followed by other European languages; whereas Facets of Buddhism and Facets of Hinduism see high performance from the Indic language models.

Perhaps more interesting, this downstream task also gives rise to circumstances in which `Topk` performance exceeded `OddOneOut`. In some cases (like Middle High German on Biblical Figures) `Topk` merely surpasses `OddOneOut` and in others (such as Gujarati on Facets of Hinduism) it is the only metric to achieve a performance score. It is likely that a complex interaction between the model and the category being evaluated on can result in either method achieving better performance, thus it is valuable to have both methods at our disposal. Since this exact relationship is still not fully understood we tune and evaluate models using both metrics and compute the harmonic mean to choose the most accurate model.

5 Related Work

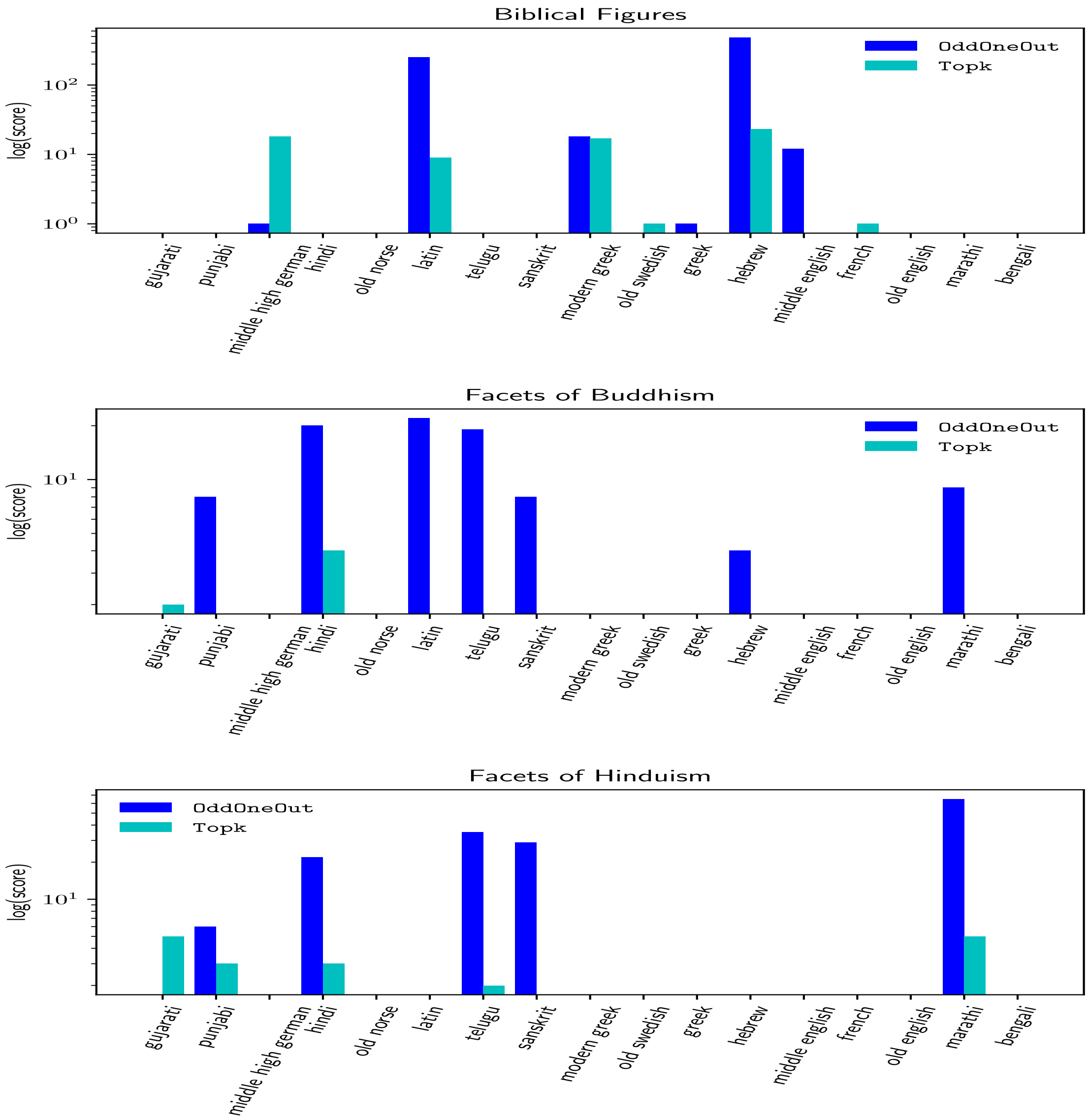

Figure 4: (**Top**) High performance on the Biblical Figures category indicates some level of biblical influence via the corpus. Interestingly, we see that greek embeddings optimized on the Modern Greek test set significantly outperformed the embeddings optimized for the Ancient Greek. This matches our intuition that things of a biblical nature have had a greater influence on Modern Greek than Ancient Greek. (**Middle**) Though many of the Indic language embeddings performed well on the Facets of Buddhism, surprisingly so did our Latin and Hebrew embeddings. This leads us to believe that some Buddhist concepts and words are shared by corpora spanning languages as diverse as Hebrew, Latin, and Hindi. At the same time, it should also be noted that Latin and Hebrew were two of the largest models trained compared to other classical languages and thus also likely benefit from greater resource richness. (**Bottom**) Similar to Figure **??** we see languages more closely related to the topic of the category achieving better performance, in this case primarily the Indic languages. Interstingly, our Bengali embeddings performed poorly on this category suggesting that the corpus did not refer heavily to the Hindu context.

	Corpus				Parameters							Scores		
	Language	Test Set	Tokens	Unique Tokens	Model	Type	Dim	Window	LR	Min Count	Lemma	`OddOneOut`	`Topk`	`Comb`
Hellenic	greek	ancient greek	37 868 209	1 877 574	w2v	cbow	40	9	-1	4	False	1140	8	
	greek	modern greek	37 868 209	1 877 574	fast	sg	90	7	-1	5	True	300	9	
Italic	latin	latin	17 777 429	470 790	w2v	cbow	50	10	-1	7	False	2748	48	
	old french	french	68 741	8343	fast	sg	250	6	-1	8	False	1	7	
Germanic	middle english	english (old)	7 048 144	314 527	fast	sg	90	5	-1	7	False	239	7	
	middle high german	german	2 090 954	60 674	fast	cbow	15	6	-1	3	False	9	19	
	old english	english (old)	104 011	33 018	fast	cbow	425	3	-1	3	True	0	1	
	old norse	icelandic	458 377	59 186	w2v	cbow	60	10	-1	3	False	968	2	
	old swedish	swedish	1 297 740	116 374	fast	sg	50	8	-1	5	False	50	1	
Indo-Aryan	bengali	bengali	5539	2323	fast	cbow	15	3	-1	4	False	0	2	
	gujarati	gujarati	1813	1140	fast	sg	80	3	-1	5	False	0	7	
	hindi	hindi	587 655	55 483	fast	cbow	45	4	-1	8	False	263	2	
	malayalam	malayalam	9235	5405	-	-	-	-	-	-	-	-	-	
	marathi	marathi	797 926	96 778	w2v	sg	400	4	-1	6	False	342	1	
	punjabi	punjabi	1 024 075	31 343	fast	sg	50	8	-1	5	False	0	1	
	sanskrit	sanskrit	4 042 204	896 480	w2v	sg	35	9	-1	10	False	1530	1	
	telugu	telugu	537 673	276 330	w2v	cbow	60	10	-1	3	False	50	0	
Semitic	classical arabic	arabic	81 306	20 493	-	-	-	-	-	-	-	-	-	
	hebrew	hebrew	41 378 460	893 512	fast	sg	30	4	-1	3	False	1098	6	

Table 1: The table above provides details for the best model trained for languages supported by CLTK. Following a tuning process, models were chosen by their `Combined Score` which is calculated as the harmonic mean of `Topk` and `OddOneOut`. It is important to note that the absolute score for our evaluation metrics are not important in and of themselves, rather they are important as indicators of a change in embedding quality that the analogy task would fail to show. To emphasize this, we have reported the raw number of correct answers for each metric. Using `OddOneOut` and `Topk` allows us to tune models trained on corpora with unique token counts in the thousands instead of the millions.

References

Wei Ai, Xuan Lu, Xuanzhe Liu, Ning Wang, Gang Huang, and Qiaozhu Mei. 2017. Untangling emoji popularity through semantic embeddings. In *Eleventh International AAAI Conference on Web and Social Media*.

Kyle P. Johnson et al. 2014–2019. Cltk: The classical language toolkit. https://github.com/cltk/cltk. DOI 10.5281/zenodo.593336.

Ziad Al-Halah, Andrew Aitken, Wenzhe Shi, and Jose Caballero. 2019. Smile, be happy:) emoji embedding for visual sentiment analysis. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0.

Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. 2013. Polyglot: Distributed word representations for multilingual nlp. *arXiv preprint arXiv:1307.1662*.

Hosein Azarbonyad, Mostafa Dehghani, Kaspar Beelen, Alexandra Arkut, Maarten Marx, and Jaap Kamps. 2017. Words are malleable: Computing semantic shifts in political and media discourse. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1509–1518.

Francesco Barbieri, Miguel Ballesteros, and Horacio Saggion. 2017. Are emojis predictable? *arXiv preprint arXiv:1702.07285*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Christian Buck, Kenneth Heafield, and Bas Van Ooyen. 2014. N-gram counts and language models from the common crawl. In *LREC*, volume 2, page 4. Citeseer.

Baitong Chen, Satoshi Tsutsui, Ying Ding, and Feicheng Ma. 2017. Understanding the topic evolution in a scientific domain: An exploratory study for the field of information retrieval. *Journal of Informetrics*, 11(4):1175–1189.

Haim Dubossarsky, Daphna Weinshall, and Eitan Grossman. 2017. Outta control: Laws of semantic change and inherent biases in word representation models. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 1136–1145.

Ben Eisner, Tim Rocktäschel, Isabelle Augenstein, Matko Bošnjak, and Sebastian Riedel. 2016. emoji2vec: Learning emoji representations from their description. *arXiv preprint arXiv:1609.08359*.

Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. *arXiv preprint arXiv:1708.00524*.

Hila Gonen, Ganesh Jawahar, Djamé Seddah, and Yoav Goldberg. 2020. Simple, interpretable and stable method for detecting words with usage change across corpora. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 538–555.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. *arXiv preprint arXiv:1802.06893*.

Sharath Chandra Guntuku, Mingyang Li, Louis Tay, and Lyle H Ungar. 2019. Studying cultural differences in emoji usage across the east and the west. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 13, pages 226–235.

Vishwani Gupta, Sven Giesselbach, Stefan Rüping, and Christian Bauckhage. 2019. Improving word embeddings using kernel PCA. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 200–208, Florence, Italy. Association for Computational Linguistics.

William L Hamilton, Jure Leskovec, and Dan Jurafsky. 2016a. Cultural shift or linguistic drift? comparing two computational measures of semantic change. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing. Conference on Empirical Methods in Natural Language Processing*, volume 2016, page 2116. NIH Public Access.

William L Hamilton, Jure Leskovec, and Dan Jurafsky. 2016b. Diachronic word embeddings reveal statistical laws of semantic change. *arXiv preprint arXiv:1605.09096*.

Austin C Kozlowski, Matt Taddy, and James A Evans. 2019. The geometry of culture: Analyzing the meanings of class through word embeddings. *American Sociological Review*, 84(5):905–949.

Vivek Kulkarni, Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. 2015. Statistically significant detection of linguistic change. In *Proceedings of the 24th International Conference on World Wide Web*, pages 625–635.

Andrey Kutuzov, Lilja Øvrelid, Terrence Szymanski, and Erik Velldal. 2018. Diachronic word embeddings and semantic shifts: a survey. *arXiv preprint arXiv:1806.03537*.

Shangsong Liang, Xiangliang Zhang, Zhaochun Ren, and Evangelos Kanoulas. 2018. Dynamic embeddings for user profiling in twitter. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1764–1773.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Andriy Mnih and Koray Kavukcuoglu. 2013. Learning word embeddings efficiently with noise-contrastive estimation. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2265–2273. Curran Associates, Inc.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA. http://is.muni.cz/publication/884893/en.

Terrence Szymanski. 2017. Temporal word analogies: Identifying lexical replacement with diachronic word embeddings. In *Proceedings of the 55th annual meeting of the association for computational linguistics (volume 2: short papers)*, pages 448–453.

Xuri Tang. 2018. A state-of-the-art of semantic change computation. *arXiv preprint arXiv:1801.09872*.

Sanjaya Wijeratne, Lakshika Balasuriya, Amit Sheth, and Derek Doran. 2017. A semantics-based measure of emoji similarity. In *Proceedings of the International Conference on Web Intelligence*, pages 646–653.

Zi Yin and Yuanyuan Shen. 2018. On the dimensionality of word embedding. In *Advances in Neural Information Processing Systems*, pages 887–898.