Fast Non-parametric Multimodal Word Embeddings

Gyanesh Gupta
gg2501@nyu.edu
New York University

Rahul Ahuja
ra3136@nyu.edu
New York University

Rohith Mukku
rm5708@nyu.edu
New York University

Abstract

Word embeddings are fundamental to Natural Language Processing - the most popular models represent each word as a numerical vector. What is lacking, however, is that the use of point estimates to represent words gives no broader picture of interactions between words other than similarity and no idea about linguistic aspects. Athiwaratkun and Wilson (2017) represent word embeddings as a bi-modal distribution where each mode represents a sense of the word. We take this idea further and allow for multimodal distributions with richer parameters that provide more insights into linguistic aspects of the language. The model inherently includes a mechanism for word sense disambiguation and the idea of "broadness" of a concept. Specifically, each sense of a word is an "effective word" and is represented as a Gaussian Distribution. Unlike previous work in the field, each word is represented as a Dirichlet Process Gaussian Mixture Model over its senses, to allow for a model that is non-parametric in the number of senses.

1 Introduction

Our motivation stems from the idea to develop a structured form for a given language, given raw text from the language. We want to focus on the big picture of building a WordNet (Feinerer and Hornik, 2020)-like dataset that can be leveraged for downstream tasks in Natural Language Processing and Understanding. Previous work has focused mainly on downstream tasks while compromising language structure, as we understand in human terms. Most research has either focused on point estimates or had constraints on the number of senses per word. We describe a fast and efficient model that captures semantic relationships between words while also capturing structured relationships. With the advances in contextual embeddings becoming mainstream,(Devlin et al., 2019), and (Peters et al., 2018), we propose adding a non-parametric soft clustering layer on top of these models.

Words mean different things within different contexts, and while contexts might be different, words only have a small finite number of meanings. The model thus needs to learn a few critical things for each word- the number of senses appropriate for word, the word vector for each word sense, covariance matrix associated with each word sense vector. The covariance matrix captures the broadness and structure of the word (sense). For example, the word "animal" would have a larger variance around the mean than the word "mammal".

We first describe a generative model for word senses - full inference on this model is possible and is neatly described in Görür and Rasmussen (2010). We train the model using the Expectation Maximization algorithm, to optimize for fast training. We represent each word as a separate Dirichlet Process Gaussian Mixture Model (DPGMM), where each Gaussian Distribution represents a word sense with its own mean vector and covariance matrix. The number of senses can be controlled by the concentration parameter α_i. α_i controls whether the senses are fine-grained or coarse-grained, smaller the alpha, larger the number of senses per word.

2 Related Work

A major breakthrough in the field of word embeddings in NLP tasks was provided by Mikolov et al. (2013) by representing words as a fixed dimensional vectors. The core idea behind word2vec was based on the distributional hypothesis, i.e.words occurring together in a large corpus of text are similar than those occurring far apart. This hypothesis was further explored in further releases of word embeddings, by using small lexemes to generate word vectors to deal with out of vocabulary words, incorporating memory based networks such

as RNNs/LSTMs or use attention based networks. In most of these approaches, the authors attempted to predict a missing word in a given "context".

However, all the above approaches give only a "point" in the n-dimensional space. To provide more insight about the word, Vilnis and McCallum (2015) treat word embeddings as a distribution, specifically a Gaussian. The idea is to fit various word contexts over a Gaussian distribution using various Bayesian methods. Theoretically, the mean of the Gaussian is the word vector or the point estimate we get from conventional approaches.

While Word2Vec are restricted to one embedding per word-sense, BERT by Devlin et al. (2019) and related approaches provide contextual embeddings, i.e. different embeddings for different contexts. This is too fine-grained and does not give a big picture about the senses, which can have different levels of granularity. Neelakantan et al. (2015) have dealt with this problem by having different embeddings for different senses. Similarly in the work by Athiwaratkun and Wilson (2017) which expand on Gaussian Word embeddings, the authors have assumed that a polysemic word usually has 2 senses. Our idea avoids this assumption and automatically finds the total number of senses associated with the word. [1]

3 Approach

3.1 Parametric Model

We start with describing a parametric model, where the number of senses for each word is known. Consider a word w_i which appears in n_i contexts c_{ij} for $1 \leq j \leq n_i$. Let w_{ik} represent the k^{th} sense of the i^{th} word, and K_i be the total number of senses for w_i. Let μ_{ik} and Σ_{ik} for $1 \leq k \leq K_i$ represent the mean and covariance matix for the k^{th} sense of w_i. Let $S_{ik} = \Sigma_{ik}^{-1}$ denote the precision matrix.

$$w_{ik} \sim \mathcal{N}(\mu_{ik}, \Sigma_{ik}) \tag{1}$$

$$w_i \sim \sum_{k=1}^{K_i} \pi_{ik} \mathcal{N}(\mu_{ik}, \Sigma_{ik}) \tag{2}$$

Let z_i be the indicator variable, a vector of size K_i, that denotes which of the K_i senses is currently being used. The natural choice for z_i given the probabilities π_i is a Multinoulli Distribution. π_i is drawn from a Dirichlet Distribution.

$$z_i|\pi_i \sim Multinoulli(\pi_{i_1}, \pi_{i_2}, \ldots, \pi_{i_{K_i}}) \tag{3}$$

$$\pi_i \sim Dirichlet(\alpha_i/K_i, \ldots, \alpha_i/K_i) \tag{4}$$

We have a Gaussian prior on mean vector μ_{ik} and a Wishart prior on the precision matrix S_{ik}. The choice of the Wishart Distribution is to maintain conjugacy with the multivariate normal distribution for each word-sense.

$$S_{ik} \sim Wishart(\beta, (\beta W)^{-1}) \tag{5}$$

$$\mu_{ik} \sim \mathcal{N}(\xi, (\rho S_{ik})^{-1}) \tag{6}$$

The hyper-parameters β, ξ, ρ, W are common across all word-senses. Note that the prior for μ_{ik} is dependent on the S_{ik}. Inference is much easier in this case as conjugacy is maintained. Without this dependency, inference is still possible, but is harder. A detailed analysis was published by (Görür and Rasmussen, 2010).

3.2 Non-parametric model

We now describe the methodology to convert the model into a non-parametric one. There are different approximations to train a non-parametric model. Since we have a finite number of data points in training, there are a finite number of non-empty clusters. (Görür and Rasmussen, 2010) add a fixed number of auxiliary empty clusters which have an equal probability of being selected. They then describe a full-inference model, updates to which are described in the Appendix A.

Full inference can be useful, but has computational overhead. We use scikit-learn, (Pedregosa et al., 2011) which uses expectation maximization to fit the model. We restrict our covariance matrices to be diagonal. The updates are described in detail in (Pedregosa et al., 2011), (Dempster et al., 1977).

3.3 Chinese Restaurant Process

The Chinese Restaurant Process is an analogy to the Dirichlet Process Model. Consider a restaurant with an infinite number of tables and some customers in some tables. When the $n+1$ th customer arrives, let's say there are K tables that are occupied. The customer sits at each table with a probability proportional to the number of people already sitting at the table or chooses a new table with probability proportional to the concentration parameter α. Let z_i be the indicator variable denoting which table the i^{th} customer chooses, n_k be the

[1] Our code for training the fast vectors is available publicly in https://github.com/gyaneshg96/word2dpgmm

number of customers at table k and $K+1$ denote a new table.

$$\mathcal{P}(z_{n+1} = K+1) = \frac{\alpha}{n+\alpha} \tag{7}$$

$$\mathcal{P}(z_{n+1} = k) = \frac{n_k}{n+\alpha} \tag{8}$$

In our setting, each context represents a customer, thus the update for indicator prior would simply replace n_k by the number of contexts representing the k^{th} sense of the word. To compute the posterior, we use

$$\mathcal{P}(z_i = k|c_i, \mu_{ik}, S_{ik}) \propto \frac{n_k}{n+\alpha_i-1}\mathcal{N}(c_{ik}|\mu_{ik}, S_{ik}) \tag{9}$$

where $n_{K_i+1} = \alpha_i$

3.4 Stick Breaking Process

In an alternative formulation, we generate the individual distributions using the "Stick Breaking Process" by Sethuraman (1994). To determine the mixture weight, we continually sample fractions from a Beta distribution, akin to continually breaking a stick. The formulation for this is:

$$\beta_i \sim Beta(1, \alpha)$$

And then use the β_i where $i \in N$ for weights like,

$$\pi_i = \beta_i \prod_{j<i}(1-\beta_j)$$

In practice, we do not consider infinite points, rather a set maximum number of clusters. Based on the parameter α, we control the actual number of clusters.

3.5 Training and Predictions

We now describe the training process. We start with a pre-trained model for contextualized word vectors, in our case we use DistilBert by Sanh et al. (2020). We divide the corpus into sub-corpora, one for each word, consisting of all the contexts the word appeared in. For every word, we then extract contextualized embeddings for each context the word appears in. Note that this step can be parallelized. Once we have contextualized word vectors, for every word, we fit a Dirichlet Process Gaussian Mixture Model, to get a (multimodal) distribution for the word and a Gaussian Distribution for each word sense.

Major advantages of using this approach are

- We can only get distributions for words we are interested in without fitting the model to the entire corpus. For example, if we only wanted to build a biological hierarchy, we would only be interested in words related to species and biological taxonomy - words like cat, tiger, felidae, mammal, fish, amoeba, animal, bird, hen, etc.

- We can fit the model to each word independently of other words (words are tied by a common model for contextualized embeddings) which is very fast as this allows for parallel processing.

Given a word-context pair, we use Equation 9 to obtain the sense of the word being used.

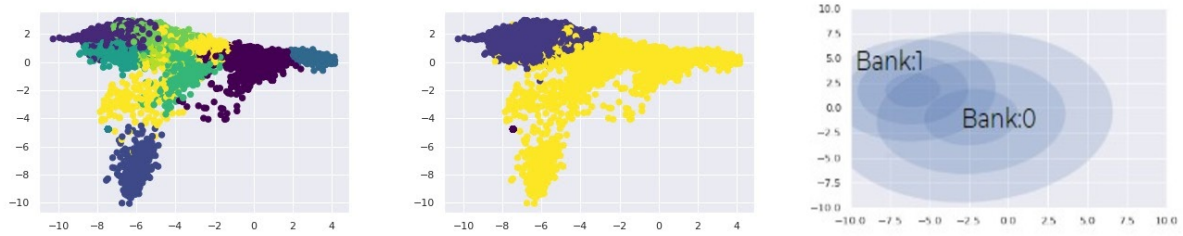
4 Experiments

4.1 Datasets

We take the UKWAC dataset by Ferraresi et al. (2008) as the corpus, containing a billion tokens. Additionally, we use the word entailment dataset by Baroni et al. (2012) for the entailment task described later on. The entailment dataset contains around 5000 positive and negative entailment pairs which we are describing in the following sections. Due to storage space and time constraints, we only use 5% of both the UKWAC and entailment dataset.

4.2 Embeddings

In the work by Athiwaratkun and Wilson (2017), the probabilistic model was trained using stochastic gradient descent while optimising on a given loss, namely KL-Divergence and probability inner product loss. However, we opted to use the pretrained contextual word embeddings as mentioned previously. Specifically, we use DistilBert by Sanh et al. (2020), a lightweight variant of BERT. We select a target word whose mixture we would want to learn and then pass maximum 10k contexts through the model. The context here is a $k = 5$ size window around each occurrence word in the corpus. For the baseline, we use the google word2vec model, and to create context aware embeddings, we take the average of all the embeddings of the word. The above procedure is repeated in parallel for multiple words.



(a) GMM for Bank (b) DPGMM for Bank (c) Gaussians formed for Bank

Figure 1: DPGMM showing better cluster detection than GMM, along with the Gaussian plots

4.3 Clustering

The next step is to cluster the obtained embeddings using the DPGMM implementation in the scikit-learn library. We tried multiple values of weight concentration parameter α, namely $0.1, 1, 100, 1000$ for some of the words, to study the affect in number of clusters formed. Our component-size was fixed to 10, which we are assuming to be the maximum for any word. We select that "diagonal" covariance setting, as for the "full" covariance type we were unable to get proper clusters. We then stored the resulting Gaussians for each word for the next task.

4.4 Word Entailment Task

To quantitatively test our embedding parameters, we perform entailment task on dataset mentioned above. The authors found KL divergence to be good metric to test this, and we employ the same. For multiple senses, we selected the sense which provided the minimum KL divergence. We came up with a threshold using KL Divergence, purely based on the maximum accuracy. We also used a simple classifier on the means and the (diagonal) covariance matrices of the word pairs to predict the task.

For a pair of Gaussian distributions p and q, with means μ_1, μ_2 and covariance matrices Σ_1, Σ_2, the KL Divergence $KL(P||Q)$ would be:

$$KL(P||Q) = \frac{1}{2}\Bigg(\log \frac{|\Sigma_2|}{|\Sigma_1|} - d + \text{tr}\{\Sigma_2^{-1}\Sigma_1\} + (\mu_2 - \mu_1)^T \Sigma_2^{-1}(\mu_2 - \mu_1) \Bigg)$$

5 Evaluation and Results

5.1 DPGMM vs GMMs

DPGMM is clearly seen as being able to detect clusters in the word embeddings for a given word. For the same number of parameter $n_{comp} = 10$, DPGMM finds the expected number of senses as shown in some of the examples. For instance, the word "bank" usually corresponds to two coarse senses, the bank related to "finance" and the bank related to "river". The above was further proven when we looked the nearest words corresponding to each sense.

In fact, DPGMM also reported a few fine-grained senses, which could be considered sub-senses of the above 2 senses. For example, we have "West Bank" which is has different meaning than the usual bank. Phrases such as "bank holidays", "World Bank" and a "bank of questions", though related to the first sense, could be thought of as a different sense.

5.2 Covariance Matrix

Intuitively, a broad word sense is something that has a "large" covariance, i.e. something that has a "flat" distribution of contexts. Quantitatively, we measure breadth of a word by the determinant of it's covariance matrix. Since the covariance matrix is positive definite, the determinant is positive. In the table 2 and 3, we have posted the log of determinant values of various words. We observe that for words related to a "topic", a broad word has more log determinant than a "niche" word.

5.3 Word Entailment Task

Our final evaluation is carried out in the word entailment task, which is how we capture the "breadth" of two different word pairs. For a pair, X, Y, we define $X \vDash Y$ or X entails Y if all instances for X are Y. For instance, we say that $aircraft$ entails $vehicle$ but cat does not entail $vehicle$.

We compare the KL Divergence of the word pair $K(P||Q)$ whose entailment truthy value needs to

Word/sense	Best Context	Nearest-Neighbours
bank:0	The forest and the eastern **bank**	river, water, bed, stones
bank:1	The **bank** accounts need to be disabled	world, money, account, england
left:0	Seeing you I though you are **left**-handed	hand, sign, road, right
left:1	There was no money **left** in the account	leave, with, stay
light:0	The lack of **light** in northern winters	sun, torch, see
light:1	**Light** hearted and jolly was him	heavy, weight, lighten, load
bug:0	The **bugs** were exterminated	insect, spider, fly, jungle
bug:1	Lots of **bugs** in production	error, resolve, code
set:0	**set** had too many duplicates	collection, list, element, of
set:1	tent was **set** up quickly	up, business, establish, up
set:2	**set** was burnt in broad daylight	studio, camera, dress

Table 1: Some examples demonstrating an application of DPGMM

WORD	LOG-DETERMINANT
BIPLANE	-6425.388
SOPRANO	-3482.880
ALGEBRA	-3343.970
PANDAS	-3542.748

Table 2: Niche Words

WORD	LOG-DETERMINANT
MACHINE	-2343.161
BUILDING	-2437.637
SOLID	-2412.539
ORGANIZATION	-2409.741

Table 3: Broad Words

be determined. Intuitively, if $K(P||Q)$ is less, then there are less extra examples for Q distribution that are not in P For our purpose we used thresholding value between the KL Divergence. Some examples are shown in the table 4. Additionally, we show results of entailment task accuracies in the table 5.

WORD PAIR	KL DIVERGENCE
CAT, ANIMAL	602.288
ANIMAL,CAT	1766.199
ANIMAL,BEVERAGE	3053.029
SHERRY,BEVERAGE	1129.993

Table 4: Word Pairs and KL Divergence

BASE-LINE	DISTIL-BERT
65.%	67.37%

Table 5: Accuracy

6 Conclusion

Our approach is somewhat successful in determining the various senses present in words. It is definitely better than simple Gaussian Mixture Models. For most contexts, we got mostly 2 senses, along with around 100 odd contexts in senses similar to the major senses. For the word "set" where we got 3 senses, in sizable amounts, which is one of the biggest positive aspects of our model.
However, some obvious senses were practically absent for some words. Such examples include a third sense for bank which relates to "curved" and the sense for "pop" related to a sound. We believe that it could largely be attributed to insufficient training data, which affects the covariance matrix.
For comparison in Word Entailment, we found the results to be better for the Distil-Bert Model as compared to our base-line, i.e. the Averaged Word Vector, as expected, as BERT is supposed to be more context aware than simple averaging. We cannot however explain why simply KL Divergence alone does not work as good predictor.

7 Future Work

- Using more data, extensive training and different types of embeddings, we can get better embeddings.
- Use better metrics than simple thresholding to predict word entailment.
- Constructing a WordNet using word-entailment is something which we can think long-term.

8 Acknowledgements

We want to thank Prof. HeHe and various TAs for their assistance in the project and also for the course instruction. We also want to thank our friend Rushab Munot, who came up with the idea behind the project.

References

Ben Athiwaratkun and Andrew Gordon Wilson. 2017. Multimodal word distributions. *ArXiv*, abs/1704.08424.

Marco Baroni, Raffaella Bernardi, Ngoc-Quynh Do, and Chung-chieh Shan. 2012. Entailment above the word level in distributional semantics. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 23–32, Avignon, France. Association for Computational Linguistics.

A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B*, 39:1–38.

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

Ingo Feinerer and Kurt Hornik. 2020. *wordnet: WordNet Interface*. R package version 0.1-15.

Adriano Ferraresi, Eros Zanchetta, Marco Baroni, and Silvia Bernardini. 2008. Introducing and evaluating ukwac , a very large web-derived corpus of english.

Dilan Görür and Carl Edward Rasmussen. 2010. Dirichlet process gaussian mixture models: Choice of the base distribution. *J. Comput. Sci. Technol.*, 25(4):653–664.

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

Arvind Neelakantan, Jeevan Shankar, Alexandre Passos, and Andrew McCallum. 2015. Efficient non-parametric estimation of multiple embeddings per word in vector space.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. Cite arxiv:1802.05365Comment: NAACL 2018. Originally posted to openreview 27 Oct 2017. v2 updated for NAACL camera ready.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

Jayaram Sethuraman. 1994. A constructive definition of the dirichlet prior. *Statistica Sinica*, 4:639–650.

Luke Vilnis and Andrew McCallum. 2015. Word representations via gaussian embedding.

A Appendices

The updates to full inference model are given as:

- Let C_i denote the set of all M_i contexts of w_i. Given a context c_{im}, we can disambiguate the context in to the relevant sense using Equation 9. Let this sense be k. Let M_{ik} denote the number of contexts belonging to sense k. Let z_{imk} denote whether the m^{th} context of the i^{th} word has sense k of the i^{th} word. Then we have:

$$\mathcal{P}(\mu_{ik}|S_{ik}, C_i) = \mathcal{N}(\frac{\rho_i\xi_i + \sum_{m-1}^{M_i} z_{imk}c_{im}}{\rho_i + M} \tag{10}$$

$$\mathcal{P}(S_{ik}|\mu_{ik}, C_i) = \mathcal{W}(\nu, V) \tag{11}$$

where

$$\begin{aligned}\nu =& \beta_i + M_i \\ V^{-1} =& \beta_i W_i \ + \sum_{m=1}^{M_i} z_{imk}(c_{im} - \mu_{ik})(c_{im} - \mu_{ik})^T\end{aligned}$$

- Updates for hyperparameters ξ_i, ρ_i and W_i have closed forms that are described in Sanh et al. (2020).

- Updates for indicators z_{imk} are based on Equation 9. We assume ζ auxiliary clusters which have no contexts and assign Z_{imk} to an auxiliary cluster with probability $\frac{\alpha_i/\zeta}{n_i+\alpha_i-1}$ and to non-empty clusters according to Equation 9.

- The posterior distributions for α_i and β_i are log-concave and can be sampled using adaptive rejection sampling as described in Görür and Rasmussen (2010).

- Since all the updates are dependent on other parameters and hyperparameters, Gibbs Sampling is the natural choise to perform inference.