COMS E6998-002: Probabilistic Modeling for Discrete Data
Lecture 3: Word Embeddings III

Instructor
David Blei
david.blei@cs.columbia.edu
(use office hours instead of email)

Scribe
Ido Rosen
ir2002@columbia.edu
(email errata to me)

19 Feb 2016

If I missed any of your comments from class below or misattributed an existing comment, it was not intentional, and I'll fix it. The following scribed notes are provided in a stream of consciousness style. If anything is missing or confusing, please let me know and I will update the notes.

0 Logistics

- If you are using GitHub, please switch to BitBucket.
- Please update your project log every time you work on your project. Be more like Tom.

1 Analogies

At the end of the previous class, we asked: Why do likelihood ratios give us analogies? Maybe this isn't the right approach. We should start from the modeling assumption that the probability of a word appearing in the context of another word is a log linear model, as follows:

$$\mathrm{P}(w|\ x^{\text{queen}}\ ) = \exp\left\{\ \sum_i x_i^{\text{queen}}\ \beta_{i,w}\ \ -\ \ a^{\text{queen}}\ \right\} \tag{1}$$

- Feature vector for "queen," x^{queen}
- Context
- We defined $\beta_{i,w}$ for analogies last lecture
- Log normalizer, $a^{\text{queen}} = \log \sum_{w'} \exp\{x^{\text{queen}} \beta_{w'}\}$

Next, let's think about the likelihood ratio between the words "queen" and "king" (as in the analogy *queen:king::female:male*:

$$\log \mathrm{P}(v|x^{\text{queen}}) - \log \mathrm{P}(v|x^{\text{king}}) \tag{2}$$

Suppose that the only difference between "queen" and "king" is that one is female and the other is male. In other words, one has the female feature vector bits set and the other has the male feature bits set. (Don't worry about latent spaces or lack of mutual exclusivity of female and male feature bits for now.) $\beta_{F,v} - \beta_{M,v} - a^{\text{queen}} + a^{\text{king}}$ Now let's remove the log normalizers and think about solving an analogy...

Note. By "the only difference", we mean "different" in the feature vectors, not different in any other way. For example, if x^{queen} and x^{king} were binary feature vectors with the only difference corresponding to the bits that *differ* between the corresponding binary features for female and male (e.g. the top two bits in the following vectors):

$$\overbrace{\begin{bmatrix}1\\0\\\hline 0\\1\\1\\0\\\vdots\end{bmatrix}}^{\text{queen}} \quad \overbrace{\begin{bmatrix}0\\1\\\hline 0\\1\\1\\0\\\vdots\end{bmatrix}}^{\text{king}} \quad \longleftrightarrow \quad \overbrace{\begin{bmatrix}1\\0\\\hline 0\\1\\1\\1\\\vdots\end{bmatrix}}^{\text{female}} \quad \overbrace{\begin{bmatrix}0\\1\\\hline 0\\1\\1\\1\\\vdots\end{bmatrix}}^{\text{male}} \tag{3}$$

(In the model, $x \in \mathbb{R}^{300}$, not $\mathbb{I}^D$, so (3) is only for demonstration.)

Consider the analogy *queen:king::woman:?* (queen is to king as woman is to ____?). What fills in the question mark? It must be a word that is identical to woman but different in the same way that queens and kings are different. In other words:

$$\begin{aligned} \beta_v{}^{\mathsf{T}} x_{\text{queen}} - \beta_v{}^{\mathsf{T}} x_{\text{king}} &= \beta_{F,v} - \beta_{M,v} && \text{(by definition)} \\ &= \beta_v{}^{\mathsf{T}} x_{\text{woman}} - \beta_v{}^{\mathsf{T}} x_{?} && \text{(objective: find ?)} \end{aligned}$$

If you ask any human, they'll probably answer $\boxed{? = \text{man}}$. (A word embedding would answer "be very very quiet, I'm hunting *wo*-bits...")

Remark. Rajesh: it's fundamentally about identifying difference in distributions of one word.

Remark. Anonymous: in the beginning of one of the papers, there was a statement that if we want linear relationships between words, we are restricted to log linear models.

Remark. The assumption that king and queen only differ in one way is also important when working with any single given v.

From the Pennington GloVe paper[9], "the relationship of these words can be examined by studying the ratio of their co-occurrence probabilities with various probe words, k." is another way of saying

that the difference in meaning is about the difference in conditional probabilities. In summary, you don't need the linearity, analogies are just about the difference of the probability distributions.

Brian asks: does this mean that if you had a huge set of analogies but no corpus, you could create the same relationship?

Question 1.1. *How do we aggregate across v (probe words)? (They set $\beta_v = 1$ across all features and proceed.)*

Question 1.2. *How do we do this with nonlinearity? (Explained previously.)*

Question 1.3. *How do we detect relations? (How do you detect what the population of relations are in a corpus?)*

For example, what about other types of relations such as part to whole, etc... The linear algebraic view is that picking these relations is basically finding an orthogonal basis that has semantic meaning, which you could do in a number of ways.

Remark. Maja suggested a couple of ways of thinking about this projection assuming the analogy is a vector, e.g. for part to whole an analogy would be a vectors such as the one pointing from Barcelona to Spain. (V^2)

Remark. Gushman(?) asks if there is a method to enforce a prior on the final distribution of words, such as for every word using a Gaussian kernel and finding the entropy of the entire space? That would imply some relations between word distributions too.

Question 1.4. *What about Bayesian analysis approaches? Are there knowledge-based and other priors that can be used? For example, non-negativity and sparsity priors?*

And for that matter, why don't the newer papers since Bengio[3] compute perplexity? (One student suggested Bengio was forward-looking in context or more supervised, where perplexity might be less appropriate in the analogy task setting.)

Remark. Maja suggested training a Gaussian mixture model (GMM) on the relation vectors, making each word an outgoing node that picks a cluster and takes that vector as output. Maybe given enough clusters, you can learn analogies.

2 Paragraph vectors

Remark. Da points out that in the `word2vec` google group (or perhaps some exclusive, secretive, Illuminati-like forum), Mikolov said he could not reproduce the paragraph vectors paper[6] results with anywhere near the performance achieved in that paper. The seminar temporarily devolves into reading out loud choice emails from the `word2vec` google group.

Assume we had a likelihood function for some word w_i and context c_i:

$$\mathcal{L} = \sum_i f(w_i, c_i) \tag{4}$$

$$f(w_i, c_i) = \log \mathrm{P}(w_i, c_i) = GLM({p_{w_i}}^{\mathsf{T}} \chi) \tag{5}$$

$$\chi = \sum_{w \in c_i} x_{c_i} \tag{6}$$

The idea behind the paragraph vector paper[6] is simple. We modify (6) above by adding a paragraph vector x_d to the model:

$$\chi = \sum_{w \in c_i} x_{c_i} + x_d \tag{7}$$

This is a mixed-effects model as shown in Figure 1 on page 4

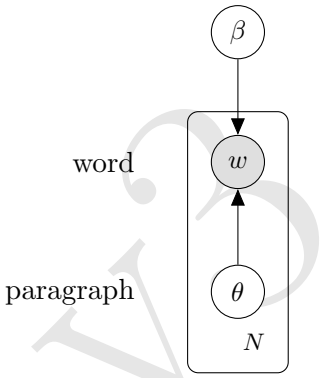

Figure 1: Paragraph vector mixed-effects model as a PGM.

As an aside, this approach suggests some kind of hierarchical model. For background, read the Gelman & Hill book on this topic.[5] Although training would harder, one could imagine a document-level, corpus-level, etc. vector in a larger hierarchical model.

Returning to the specifics of the paragraph vector model, the same paragraph vector is used for all the words in the paragraph. It captures what's common to the words in the paragraph, rather than just the context of $\pm n$ words. Intuitively, this seems prone to overfitting. (Note also in the model above that the sum is inside the exponent.) However, intuitively it also might improve accuracy since paragraphs typically correspond to a complete thought (sometimes even a single topic), semantically, whereas n-words of context may not, especially with small n.

For example, consider the problem of finding the word to fill in the bank in the sentence: "She ____ the ball." In word embeddings, we don't necessarily know how to fill this in: kicks, hit, throws, etc. could all similarly qualify. The paragraph vector model might help us pick the right verb for the situation given the paragraph as context, for example "hit" might make more sense in a paragraph about baseball. With a regular word embedding, kicked and hit are equally liked. The paragraph

vector in the ideal case will lean us to baseball, making hit more likely.

A couple of questions came up:

Question 2.1. *Is this model additive?* $\exp(a+b)$

♦♦♦

Question 2.2. *Do β and θ depend on eachother?*

In the second model, Le and Mikolov (i) change their model to longer use logistic regression, and (ii) change the normalizer. The second model removed context, just fitting paragraph vectors. This means that in the second model we have $\chi = x_d$. This is just a random effects model.

Remark. Maybe it works? It's not clear, in part because according to Mikolov as of November 2015 on the `word2vec` Google group, there is a 9% error rate in the reproduced version, rather than a $< 8\%$ error rate from the paper. Mikolov also said on those forums that they didn't shuffle/randomize the order of the minibatches, which could explain the difference.

Remark. Paragraph vectors relate to the slow random walk discourse vectors in the Arora RAND-WALK paper[1].

3 Skip-gram with negative sampling as matrix factorization

Mutual information on random variables X and Y is defined as:

$$MI(X,Y) = \mathbb{E}\left[\log \mathrm{P}(x,y) - \log \mathrm{P}(x) - \log \mathrm{P}(y)\right] \tag{8}$$

If X and Y are conditionally independent, then $MI(X,Y) = \log 1 = 0$. (Mutual information also has some other nice properties such as non-negativity and symmetry.) If you are not familiar with information theory, read Cover & Thomas[4].

Most people agreed that this was the best paper of the three we read this week. It even had a nicer explanation of the skip-gram model with negative sampling than the original skip-gram paper, in your humble scribe's opinion.

The objective function in the Levy Goldberg paper[7] is as follows:

$$\mathcal{L} = \sum_i \log \sigma({\rho_{w_i}}^\mathsf{T} x_{c_i}) \; + k \; \mathbb{E}_C \left[\log \sigma(-{\rho_{w_i}}^\mathsf{T} x_C) \right] \tag{9}$$

- Log probability that w_i is in the data
- Expectation over context
- Log probability that w_i is *not* in the data

This is close to a logistic regression, but not quite. Let's think about the two random variables W and C, where W is a randomly chosen word, and C is a randomly chosen context. The mutual information, which measures how dependent these variables are, i.e. how much knowing the context

tells me about the word, is $MI(W,C) = \sum_c \sum_w PMI(w,c)$ with PMI meaning pointwise mutual information when $k = 1$ in the model above. In the context of this paper, we thus have the following expression for mutual information of words and contexts:

$$MI(W,C) = \sum_w \sum_c \log\left(\frac{\#(w,c)n}{\#(w)\#(c)}\right) \tag{10}$$

$$= {\rho_w}^\top x_c \qquad \text{when k=1} \tag{11}$$

This presents another way of thinking about word vectors: as vectors that help us understand mutual information, a measurement of dissimilarity or dependence.

Remark. Anonymous student: If you were just doing max likelihood estimation (MLE) for the logistic regression, you wouldn't end up with pointwise mutual information. In the logistic regression MLE the specific context doesn't appear in the second term (the expectation over contexts), whereas in this model it does. So, this regularizes by saying that we won't be overconfident in having *not* seen the word.

Question 3.1. *How do we choose k?* $(PMI(w,c) - \log(1/k))$

By cross-validation, obviously. With $k = 1$ we get the mutual information. Let's stick to $k = 1$.

The paper considers positive pointwise mutual information, calculating $MI(W,C)$ by using $PPMI$:

$$PPMI(w,c) = \max(PMI(w,c), 0) \tag{12}$$

This is the first of two hacks: (i) ignore anything below zero with PPMI to avoid negative infinities, and (2) ?.

Remark. Tom: The idea of positive interactions moves beyond analogies to metaphors, which is interesting. For example, "my love is a red red rose" vs "my love is not a black forest." It's interesting that it's now a different space: metaphorical rather than analogical.

Equation 13 in the paper, where they multiple by the square root of the singular values, is better for using SVD for word vectors ($W = U \cdot \sqrt{\sigma}$). This works better than existing methods. Run SVD, set word and context vector to these.

We floated the idea in class of using Bayesian optimization to choose k.

4 Linear algebraic structure of word embeddings

The discussion around polysemy began with an experiment where multiple words (e.g. cat, dog) were replaced with the same token (e.g. dog) to induce artificial polysemy. Let's think probabilistically: we can approach polysemy by imagining a word vector as a mixture, so that we have a

mixture model underneath each word vector. *Polysemy* is when one word may have multiple meanings. A specific type of polysemy came up towards the end of the discussion around *contronyms* (words that have opposite meanings in different contexts; e.g. sanction), particularly in the law.

Arora[2] uses the adea of a word vector as a linear combination and does something similar to a mixture model:

$$v_w = \sum_{j=1}^{n} \alpha_{w,j} A_j + \eta_w \tag{13}$$

Remark. Aren't these just topic models in sheep's clothing?

This model is really just learning recurring patterns of meaning features. Each word is a combination of these basic patterns, which is why they start looking like topics. It's close to topics, but not quite.

Remark. Brian asks: Has anyone quantified how big of a problem polysemy is actually? Does it matter that much?

When you plot these vectors in lower dimensional space (e.g. t-SNE) you might see "bank" between all the finance (money, etc.) and the river (beaver, etc.) words.

Maja suggested the idea of projections into different meaning subspaces.

Brian liked the Arora paper, bucking the trend: He mentioned that you could use polysemy to detect puns or double entendres, perhaps even build and innuendo generator.

Another student asked if there was a gold standard dataset for polysemy. There does not seem to be, but contronyms would be a good gold standard potentially. This began a discussion around contronyms such as "sanction."

5 Closing remarks

Are word embedding problems hard enough? Or, are we just reading papers knocking off low hanging fruit? Can't we solve most of these problems with Naive Bayes or MLE? Dave quips "I spent 5 years in grad school trying to beat Naive Bayes."

Jaan mentioned that perhaps there are music applications for a similar approach to word embeddings, such as projections across octaves/scales.

Next week we'll move on from word embeddings to matrix/tensor factorization.

A References

[1] S. Arora, Y. Li, Y. Liang, T. Ma, and A. Risteski. Random walks on context spaces: Towards an explanation of the mysteries of semantic word embeddings. *arXiv preprint*, arXiv:abs/1502.03520v5, Oct. 2015. URL http://arxiv.org/abs/1502.03520.

[2] S. Arora, Y. Li, Y. Liang, T. Ma, and A. Risteski. Linear algebraic structure of word senses, with applications to polysemy. *arXiv preprint*, arXiv:abs/1601.03764, Jan. 2016. URL http://arxiv.org/abs/1601.03764.

[3] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155, 2003. URL http://www.jmlr.org/papers/volume3/bengio03a/bengio03a.pdf.

[4] T. M. Cover and J. A. Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[5] A. Gelman and J. Hill. *Data analysis using regression and multilevel/hierarchical models*. Cambridge University Press, 2006.

[6] Q. Le and T. Mikolov. Distributed representations of sentences and documents. In *International Conference on Machine Learning*, volume 31, pages 1188–1196, 2014. URL http://jmlr.org/proceedings/papers/v32/le14.html.

[7] O. Levy and Y. Goldberg. Neural word embedding as implicit matrix factorization. In *Advances in Neural Information Processing Systems*, volume 27, pages 2177–2185. 2014. URL http://papers.nips.cc/paper/5477-neural-word-embedding-as-implicit-matrix-factorization.pdf.

[8] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, volume 26, pages 3111–3119, 2013. URL http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf.

[9] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014. URL http://www-nlp.stanford.edu/pubs/glove.pdf.