ON MODELING HIERARCHICAL DATA VIA ENCAPSULATION OF PROBABILITY DENSITIES

Anonymous authors
Paper under double-blind review

ABSTRACT

By representing words with probability densities rather than point vectors, probabilistic word embeddings can capture rich and interpretable semantic information and uncertainty (Vilnis & McCallum, 2014; Athiwaratkun & Wilson, 2017). For example, such embeddings trained on an unlabelled corpus can represent lexical entailment, where the learned distribution for a general concept such as “animal” can contain the distributions for more specific words such as “dog” or “cat”. However, for some words such as “mammal”, the entailment signal is often weak since “mammal” is usually not used in place of “dog” or “cat” in natural sentences. In this paper, we develop the use of density representations to specifically model hierarchical data. We introduce simple yet effective loss functions and distance metrics, as well as graph-based schemes to select negative samples to better learn hierarchical probabilistic representations. Our methods outperform the original methodology proposed by Vilnis & McCallum (2014) by a significant margin and provide state-of-the-art performance on the WORDNET hypernym relationship prediction task and the challenging HYPERLEX lexical entailment dataset.

1 INTRODUCTION

Learning feature representations of natural data such as text and images has become increasingly important for understanding real-world concepts. These representations are useful for many tasks, ranging from semantic understanding of words and sentences (Mikolov et al., 2013; Kiros et al., 2015), image caption generation (Vinyals et al., 2015), textual entailment prediction (Rocktäschel et al., 2015), to language communication with robots (Bisk et al., 2016).

Meaningful representations of text and images capture visual-semantic information, such as hierarchical structure where certain entities are abstractions of others. For instance, an image caption “A dog and a frisbee” is an abstraction of many images with possible lower-level details such as a dog jumping to catch a frisbee or a dog sitting with a frisbee (Figure 1a). A general word such as object is also an abstraction of more specific words such as house, pool. Recent work by Vendrov et al. (2015) proposes learning such asymmetric relationships with order embeddings – vector representations of non-negative coordinates with partial order structure. These embeddings are shown to be effective for word hypernym classification, image-caption ranking (Vendrov et al., 2015) and textual entailment.

Another recent line of work involves using probability distributions as rich feature representations that can capture uncertainty and nuances of words such as Gaussian word embeddings (Vilnis & McCallum, 2014), or glean disentangled semantic representations via multimodal densities (Athiwaratkun & Wilson, 2017). Probability distributions are also natural at capturing orders and are suitable for tasks that involve hierarchical structures. An abstract entity such as animal that can represent specific entities such as insect, dog, bird corresponds to a broad distribution, encapsulating the distributions for these specific entities. For example, the distribution for insect is more concentrated than for animal, with a high density occupying a small volume in space. Such entailment patterns can be observed from density word embeddings trained on a natural corpus (Vilnis & McCallum, 2014; Athiwaratkun & Wilson, 2017). However, the nature of word occurrences in a text corpus often results in a weak signal for learning some hypernym relationships. For instance, it is fairly common to observe sentences “Look at the cat”, or “Look at the dog”, but not “Look at the mammal”. Therefore, due to the way we typically express natural language, it is unlikely that the word “mammal” would
be learned, with existing methodology, as a distribution that encompasses both “cat” and “dog”, since “mammal” rarely occurs in similar contexts.

The goal of this paper is to investigate the use of probability densities to model the hierarchical structure in a supervised corpus. In particular, we adopt the Gaussian representation of entities, similar to Gaussian word embeddings by Vilnis & McCallum (2014). However, we proposed a different training approach that show a superior performance over the training approach by Vilnis & McCallum (2014) on supervised data. In particular, we adopt a new form of loss function and use a thresholded divergence-based penalty function that do not further penalize the model if there is a desirable degree of density encapsulation. In addition, we introduced a new graph-based scheme to select negative samples to contrast the true relationship pairs during training. We also demonstrate that initializing the right variance scale is highly important for modeling hierarchical data via distributions, allowing the model to exhibit meaningful encapsulation orders. The variance scale is highly important, due to the fact that a great deal of information in density embeddings is captured in the variance, unlike in the traditional word distributions where the mean vectors often capture most of the semantics (Vilnis & McCallum, 2014; Athiwaratkun & Wilson, 2017).

In Section 2, we introduce the background for Gaussian embeddings and vector order embeddings. We introduce our training methodology in Section 3 and describe the experiment details in Section 4. We offer a qualitative evaluation of the model in Section 4.3, where we show the visualization of the density encapsulation behavior. We show quantitative results on WORDNET Hypernym prediction task in Section 4.2 and a graded entailment dataset HYPERLEX in Section 4.4, our models achieve the state-of-the-art results. We will make the training and evaluation code publicly available upon publication.

![Figure 1](image.png)

Figure 1: (a) Captions and images in the visual-semantic hierarchy. (b) Vector order embedding (Vendrov et al., 2015) where specific entities have higher coordinate values. (c) Density order embedding where specific entities correspond to concentrated distributions encapsulated in broader distributions of general entities.

## 2 Background and Related Work

### 2.1 Gaussian Embeddings

Vilnis & McCallum (2014) was the first to propose using probability densities as word embeddings. In particular, each word is modeled as a Gaussian distribution, where the mean vector represents the semantics and the covariance describes the uncertainty or nuances in the meanings. These embeddings are trained on a natural text corpus by maximizing the similarity between words that are in the same local context of sentences. Given a word \( w \) with a true context word \( c_p \) and a randomly sampled word \( c_n \) (negative context), Gaussian embeddings are learned by minimizing the rank objective in Equation 1, which pushes the similarity of the true context pair \( E(w, c_p) \) above that of the negative context pair \( E(w, c_n) \) by a margin \( m \).

\[
L_m(w, c_p, c_n) = \max(0, m - E(w, c_p) + E(w, c_n))
\]  

(1)

The similarity score \( E(u, v) \) for words \( u, v \) can be either \( E(u, v) = -\text{KL}(f_u, f_v) \) or \( E(u, v) = (f_u, f_v)_2 \) where \( f_u, f_v \) are the distributions of words \( u \) and \( v \), respectively. The Gaussian word embeddings contain rich semantic information and performs competitively in many word similarity benchmarks.
The true context word pairs \((w, c_p)\) are obtained from natural sentences in text corpus such as Wikipedia. In some cases, specific words can be replaced by a general word in similar context. For instance, “I love cats” or “I love dogs” can be replaced with “I love animals”. Therefore, the trained word embeddings exhibit lexical entailment patterns where specific words such as dog and cat are concentrated distributions that are encompassed by a more dispersed distribution of animal, a word that cat and dog entail. The broad distribution of a general word agrees with the distributional informativeness hypothesis proposed by Santus et al. which says that generic words occur in more general contexts than specific ones.

However, some word entailment pairs have weak density encapsulation pattern due to the nature of word diction. For instance, even though “dog” and “cat” both entail “mammall”, it is rarely the case that we observe a sentence “I have a mammall” as opposed to “I have a cat” in a natural corpus; therefore, after training density word embeddings on word occurrences, encapsulation of some true entailment instances do not occur.

### 2.2 Partial Orders and Vector Order Embeddings

We describe partial order and the concept of order embeddings proposed by Vendrov et al. (2015), which is highly related to our model.

A partial order over a set of points \(X\) is a binary relation \(\preceq\) such that for \(a, b, c \in X\), the following properties hold: (1) \(a \preceq a\) (reflexivity); (2) if \(a \preceq b\) and \(b \preceq a\) then \(a = b\) (antisymmetry); and (3) if \(a \preceq b\) and \(b \preceq c\) then \(a \preceq c\) (transitivity). An example of a partially ordered set is a set of points in a tree where \(a \preceq b\) means \(a\) is a child node of \(b\). This concept has applications in natural data such as lexical entailment. For words \(a\) and \(b\), \(a \preceq b\) means that every instance of \(a\) is an instance of \(b\), or we can say that \(a\) entails \(b\). We also say that \((a, b)\) has a hypernym relationship where \(a\) is a hypernym of \(b\) and \(b\) is a hypernym of \(a\). This relationship is asymmetric since \(a \preceq b\) does not necessarily imply \((b \preceq a)\). For instance, aircraft \(\preceq\) vehicle but it is not true that vehicle \(\preceq\) aircraft.

An order-embedding is a function \(f : (X, \preceq_X) \to (Y, \preceq_Y)\) where \(a \preceq_X b\) if and only if \(f(a) \preceq_Y f(b)\). Vendrov et al. (2015) proposes to learn the embedding \(f\) on \(Y = \mathbb{R}^N_+\) where all coordinates are non-negative. Under \(\mathbb{R}^N_+\), there exists a partial order relation called the reversed product order on \(\mathbb{R}^N_+\): \(x \preceq y\) if and only if \(\forall i, x_i \geq y_i\). That is, a point \(x\) entails \(y\) if and only if all the coordinate values of \(x\) is higher than \(y\)’s. The origin represents the most general entity at the top of the order hierarchy and the points further away from the origin become more specific. Figure 1b demonstrates the vector order embeddings on \(\mathbb{R}^N_+\). We can see that since insect \(\preceq\) animal and animal \(\preceq\) organism, we can infer directly from the embedding that insect \(\preceq\) organism (orange line, diagonal line). To learn the embeddings, Vendrov et al. (2015) proposes a penalty function \(E(x, y) = \|\max(0, y - x)\|^2\) for a pair \(x \preceq y\) which has the property that it is positive if and only if the order is violated.

### 3 Methodology

In Section 3.1, we describe the partial orders that can be induced by density encapsulation. Section 3.2 describes our training approach that softens the notion of strict encapsulation with a viable penalty function.

#### 3.1 Strict Encapsulation Partial Orders

A partial order on probability densities can be obtained by the notion of encapsulation. That is, a density \(f\) is more specific than a density \(g\) if \(f\) is encompassed in \(g\). The degree of encapsulation can vary, which gives rise to multiple order relations. We define an order relation \(\preceq_\eta\) for \(\eta \geq 0\) where \(\eta\) indicates the degree of encapsulation required for one distribution to entail another. More precisely, for distributions \(f\) and \(g\),

\[
f \preceq_\eta g \iff \{x : f(x) > \eta\} \subseteq \{x : g(x) > \eta\}.
\]

Note that \(\{x : f(x) > \eta\}\) is a set where the density \(f\) is greater than the threshold \(\eta\). The relation in Equation 2 says that \(f\) entails \(g\) if and only if the set of \(g\) contains that of \(f\). In Figure 2, we depict two Gaussian distributions with different mean vectors and covariance matrices. Figure 2 (left) shows the density values of distributions \(f\) (narrow, blue) and \(g\) (broad, orange) and different threshold
levels. Figure 2 (right) shows that different η’s give rise to different partial orders. For instance, we observe that neither \( f \preceq_{n_1} g \) nor \( g \preceq_{n_1} f \) but \( f \preceq_{n_3} g \).

3.2 Soft Encapsulation Orders

A plausible penalty function for the order relation \( f \preceq_{\eta} g \) is a set measure on \( \{ x : f(x) > \eta \} - \{ x : g(x) > \eta \} \). However, this is difficult to calculate for most distributions, including Gaussians. Instead, we use simple penalty functions based on asymmetric divergence measures between probability densities. Divergence measures \( D(\cdot || \cdot) \) have a property that \( D(f || g) = 0 \) if and only if \( f = g \). Using \( D(\cdot || \cdot) \) to represent order violation is undesirable since the penalty should be 0 if \( f \neq g \) but \( f \preceq g \). Therefore, we propose using a thresholded divergence

\[
d_{\gamma}(f, g) = \max(0, D(f || g) - \gamma),
\]

which can be zero if \( f \) is properly encapsulated in \( g \). We discuss the effectiveness of using divergence thresholds in Section A.2.1.

We note that by using \( d_{\gamma}(\cdot, \cdot) \) as a violation penalty, we no longer have the strict partial order. In particular, the notion of transitivity in a partial order is not guaranteed. For instance, if \( f \preceq g \) and \( g \preceq h \), our density order embeddings would yield \( d_{\gamma}(f, g) = 0 \) and \( d_{\gamma}(g, h) = 0 \). However, it is not necessarily the case that \( d_{\gamma}(f, h) = 0 \) since \( D(f || h) \) can be greater than \( \gamma \). This is not a drawback since high value of \( D(f || h) \) reflects that the hypernym relationship is not “direct”, requiring many edges from \( f \) to \( h \) in the hierarchy. The extent of encapsulation contains useful entailment information, as demonstrated in Section 4.4 where our model scores highly correlate with the annotated scores of a challenging lexical entailment dataset and achieves state-of-the-art result.

Another property, antisymmetry, does not strictly hold since if \( d_{\gamma}(f, g) = 0 \) and \( d_{\gamma}(g, f) = 0 \) does not imply \( f = g \). However, in this situation, it is necessary that \( f \) and \( g \) overlap significantly if \( \gamma \) is small. Due to the fact that the \( d_{\gamma}(\cdot, \cdot) \) does not strictly induce a partial order, we refer to this model as soft density order embeddings or simply density order embeddings.

3.3 Divergence Measures

3.3.1 Asymmetric Divergence

Kullback-Leibler (KL) Divergence The KL divergence is an asymmetric measure of the difference between probability distributions. For distributions \( f \) and \( g \), \( \text{KL}(g || f) \equiv \int g(x) \log \frac{g(x)}{f(x)} \, dx \) imposes a high penalty when there is a region of points \( x \) such that the density \( f(x) \) is low but \( g(x) \) is high. An example of such a region is the area on the left of \( f \) in Figure 2. This measure penalizes the situation where \( f \) is a concentrated distribution relative to \( g \); that is, if the distribution \( f \) is encompassed by \( g \), then the KL yields high penalty. For \( d \)-dimensional Gaussians \( f = \mathcal{N}_d(\mu_f, \Sigma_f) \) and \( g = \mathcal{N}_d(\mu_g, \Sigma_g) \),

\[
2D_{KL}(f || g) = \log(\text{det}(\Sigma_g)/\text{det}(\Sigma_f)) - d + \text{tr}(\Sigma_g^{-1}\Sigma_f) + (\mu_f - \mu_g)^T \Sigma_g^{-1}(\mu_f - \mu_g)
\]

Rényi \( \alpha \)-Divergence is a general family of divergence with varying scale of zero-forcing penalty (Renyi, 1961). Equation 4 describes the general form of the \( \alpha \)-divergence for \( \alpha \neq 0, 1 \) (Friedrich Liese, 1987). We note that for \( \alpha \to 0 \) or 1, we recover the KL divergence and the reverse KL divergence; that is, \( \lim_{\alpha \to 1} D_\alpha(f || g) = \text{KL}(f || g) \) and \( \lim_{\alpha \to 0} D_\alpha(f || g) = \text{KL}(g || f) \) (Pardo, 2006). The \( \alpha \)-divergences are asymmetric for all \( \alpha \)'s, except for \( \alpha = 1/2 \).

\[
D_\alpha(f || g) = \frac{1}{\alpha(\alpha - 1)} \log \left( \int \frac{f(x)^\alpha}{g(x)^{\alpha - 1}} \, dx \right)
\]
We note that this loss function is different than the rank-margin loss introduced in the original paper. For flat data such as words in a text corpus, negative samples from the dataset. Minimizing this function (Equation 7) is equivalent to minimizing the penalty (Baroni et al., 2012).

To learn our density embeddings, we use a loss function similar to that of Socher et al. (2013) and Vendrov et al. (2015). The parameter $\alpha$ controls the degree of zero forcing where minimizing $D_{\alpha}(f||g)$ for high $\alpha$ results in $f$ being more concentrated to the region of $g$ with high density. For low $\alpha$, $f$ tends to be mass-covering, encompassing regions of $g$ including the low density regions. Recent work by Li & Turner (2016) demonstrates that different applications can require different degrees of zero-forcing penalty.

3.3.2 Symmetric Divergence

Expected Likelihood Kernel The expected likelihood kernel (ELK) (Jebara et al., 2004) is a symmetric measure of affinity, define as $K(f, g) = \langle f, g \rangle_{\mathcal{H}}$. For two Gaussians $f$ and $g$,

$$2 \log \langle f, g \rangle_{\mathcal{H}} = - \log \det(\Sigma_f + \Sigma_g) - d \log(2\pi) - (\mu_f - \mu_g)^T (\Sigma_f + \Sigma_g)^{-1} (\mu_f - \mu_g)$$

(6)

Since this kernel is a similarity score, we use its negative as our penalty. That is, $D_{\text{ELK}}(f||g) = -2 \log \langle f, g \rangle_{\mathcal{H}}$. Intuitively, the asymmetric measures should be more successful at training density order embeddings. However, a symmetric measure can result in the encapsulation order as well since a general entity often has to minimize the penalty with many specific elements and consequently ends up having a broad distribution to lower the average loss. The expected likelihood kernel is used to train Gaussian and Gaussian Mixture word embeddings on a large text corpus (Vilnis & McCallum, 2014; Athiwaratkun & Wilson, 2017) where the model performs well on the word entailment dataset (Baroni et al., 2012).

3.4 Loss Function

To learn our density embeddings, we use a loss function similar to that of Socher et al. (2013) and Vendrov et al. (2015). Minimizing this function (Equation 7) is equivalent to minimizing the penalty between a true relationship pair $(u, v)$ where $u \preceq v$, but pushing the penalty to be above a margin $m$ for the negative example $(u', v')$ where $u' \not\preceq v'$:

$$\sum_{(u, v) \in D} d(u, v) + \max\{0, m - d(u', v')\}$$

(7)

We note that this loss function is different than the rank-margin loss introduced in the original Gaussian embeddings (Equation 1). Equation 7 aims to reduce the dissimilarity of a true relationship pair $d(u, v)$ with no constraint, unlike in Equation 1, which becomes zero if $d(u, v)$ is above $d(u', v')$ by margin $m$.

3.5 Selecting Negative Samples

In many embedding models such as word2vec or Gaussian embeddings (Mikolov et al., 2013; Vilnis & McCallum, 2014), negative samples are often used in the training procedure to contrast with true samples from the dataset. For flat data such as words in a text corpus, negative samples are selected randomly from a unigram distribution. We propose new graph-based methods to select negative samples that are suitable for hierarchical data, as demonstrated by the improved performance of our density embeddings. In our experiments, we use various combinations of the following methods.

Method S1: A simple negative sampling procedure used by Vendrov et al. (2015) is to replace a true hypernym pair $(u, v)$ with either $(u, v')$ or $(u', v)$ where $u', v'$ are randomly sampled from a uniform distribution of vertices. Method S2: We use a negative sample $(v, u)$ if $(u, v)$ is a true relationship pair. The motivation is due to the fact that it is important to make $D(v||u)$ higher than $D(u||v)$ in order to distinguish the directionality of density encapsulation. Method S3: It is important to increase the divergence between neighbor entities that do not entail each other. Let $A(w)$ denote all descendants of $w$ in the training set $D$, including $w$ itself. We first randomly sample an entity $w \in D$ that has at least 2 descendants and randomly select a descendant $u \in A(w) - \{w\}$. Then, we randomly select an entity $v \in A(w) - A(u)$ and use the random neighbor pair $(v, u)$ as a negative...
sample. Note that we can have \( u \preceq v \), in which case the pair \((v, u)\) is a reverse relationship. Method S4: Same as S3 except that we sample \( v \in A(w) - A(u) - \{w\} \). This excludes the possibility of drawing \((w, u)\).

4 Experiments

This section describes the training details to learn the hypernym hierarchy on the WORDNET dataset (Miller, 1995) with the density order embeddings.

4.1 Training Details

We have a similar data setup to the experiment by Vendrov et al. (2015) where we use the transitive closure of WordNet noun hypernym relationships which contains 82,115 synsets and 837,888 hypernym pairs from 84,427 direct hypernym edges. We obtain the data using the WORDNET API of NLTK version 3.2.1 (Loper & Bird, 2002). We use 4000 hypernym relationships for the validation set and another 4000 for the test set.

We use \( d \)-dimensional Gaussian distributions with diagonal covariance matrices. We use \( d = 50 \) as the default dimension and analyze the results using different \( d \)'s in Section A.2.4. We initialize the mean vectors to have a unit norm and normalize the mean vectors in the training graph. We initialize the diagonal variance components to be all equal to \( \beta \) and optimize on the unconstrained space of \( \log(\Sigma) \). We discuss the important effects of the initial variance scale in Section A.2.2.

We use a minibatch size of 500 true hypernym pairs and use varying number of negative hypernym pairs, depending on the negative sample combination proposed in Section 3.5. We discuss the results for many selection strategies in Section 4.4. We also experiment with multiple divergence measures \( D(||\cdot||) \) described in Section 3.3. In our main results, we use \( D(||\cdot||) = D_{KL}(||\cdot||) \) unless stated otherwise. Section A.2.5 discuss the results using the \( \alpha \)-divergence family with varying degrees of zero-forcing parameter \( \alpha \)'s. We use the Adam optimizer (Kingma & Ba, 2014) and train our model for at most 20 epochs. For each energy function, we tune the hyperparameters on grids. The hyperparameters are the loss margin \( m \), the initial variance scale \( \beta \), and the energy threshold \( \gamma \). We evaluate the results by computing the penalty on the validation set to find the best threshold for binary classification, and use this threshold to perform prediction on the test set. Section A.1 describes the hyperparameters for all our models.

4.2 Hypernym Prediction

We show the prediction accuracy results on the test set of WORDNET hypernyms in Table 1. We compare our results with vector order-embeddings (VOE) by Vendrov et al. (2015) where the model details are explained in Section 2.2. Another important baseline is the transitive closure, which requires no learning and classifies if a held-out edge is a hypernym relationship by determining if it is in the union of the training edges. word2gauss and word2gauss† are the Gaussian embeddings trained using the loss function in Vilnis & McCallum (2014) (Equation 1) where word2gauss is the result reported by Vendrov et al. (2015) and word2gauss† is the best performance of our replication (see Section A.2.3 for more details). Our density order embedding (DOE) outperforms the implementation by Vilnis & McCallum (2014) significantly; this highlights the fact that a different approach to train Gaussian representations might be required for a different task.

Our method also outperforms the vector order embeddings (VOE). We also include the results for a 2-dimensional Gaussian embedding trained for the purpose of visualization (Section 4.3). Surprisingly, the performance is very strong, beating the transitive closure and other baselines except VOE while only having 4 parameters: 2 from 2-dimensional \( \mu \) and another 2 from the diagonal \( \Sigma \). The results using a symmetric measure also outperforms the baselines but has a slightly lower accuracy than the asymmetric model.

Figure 3 offers an explanation as to why our density order embeddings might be easier to learn, compared to the vector counterpart. In certain cases such as fitting a general concept entity to the embedding space, we simply need to adjust the distribution of entity to be broad enough to encompass all other concepts. In the vector counterpart, it might be required to shift many points further from the origin to accommodate entity to reduce cascading order violations.
Figure 3: (Left) Adding a concept entity to vector order embedding (Right) Adding a concept entity to density order embedding

Table 1: Classification accuracy on hypernym relationship test set from WordNet.

<table>
<thead>
<tr>
<th>Method</th>
<th>Test Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>transitive closure</td>
<td>88.2</td>
</tr>
<tr>
<td>word2gauss</td>
<td>86.6</td>
</tr>
<tr>
<td>word2gauss†</td>
<td>88.6</td>
</tr>
<tr>
<td>VOE (symmetric)</td>
<td>84.2</td>
</tr>
<tr>
<td>VOE</td>
<td>90.6</td>
</tr>
<tr>
<td>DOE (ELK)</td>
<td>92.1</td>
</tr>
<tr>
<td>DOE (KL, reversed)</td>
<td>83.2</td>
</tr>
<tr>
<td>DOE (KL)</td>
<td><strong>92.3</strong></td>
</tr>
<tr>
<td>DOE (KL, (d = 2))</td>
<td>89.2</td>
</tr>
</tbody>
</table>

4.3 Qualitative Analysis

For qualitative analysis, we additionally train a 2-dimensional Gaussian model for visualization. Our qualitative analysis shows that the encapsulation behavior can be observed in the trained model. Figure 4 demonstrates the ordering of synsets in the density space. Each ellipse represents a Gaussian distribution where the center is given by the mean vector \(\mu\) and the major/minor axes are given by the diagonal standard deviations \(\Sigma\), scaled by 300 and 30 for \(x\) and \(y\) axis for visibility.

Most hypernym relationships exhibit the encapsulation behavior where the hypernym encompasses the synset that entails it. For instance, the distribution of \(\text{whole.n.02}\) is subsumed in the distribution of \(\text{physical_entity.n.01}\). Note that \(\text{location.n.01}\) is not entirely encapsulated by \(\text{physical_entity.n.01}\) under this visualization. However, we can still predict which entity should be the hypernym among the two since the KL divergence of one given another would be drastically different. This is because a large part of \(\text{physical_entity.n.01}\) has considerable density at the locations where \(\text{location.n.01}\) has very low density. This causes \(\text{KL}(\text{physical_entity.n.01} \mid | \text{location.n.01})\) to be very high (5103) relative to \(\text{KL}(\text{location.n.01} \mid | \text{physical_entity.n.01})\) (206). Table 2 shows the KL values for all pairs where we note that the numbers are from the full model \((d = 50)\).

Another interesting pair is \(\text{city.n.01} \leq \text{location.n.01}\) where we see the two distributions have very similar contours and the encapsulation is not as distinct. In our full model \(d = 50\), the distribution of \(\text{location.n.01}\) encompasses \(\text{city.n.01}'s\), indicated by low \(\text{KL}(\text{city.n.01} \mid | \text{location.n.01})\) but high \(\text{KL}(\text{location.n.01} \mid | \text{city.n.01})\).

Figure 4 (Right) demonstrates the idea that synsets on the top of the hypernym hierarchy usually have higher “volume”. A convenient metric that reflects this quantity is \(\log \det(\Sigma)\) for a Gaussian distribution with covariance \(\Sigma\). We can see that the synset, \(\text{physical_entity.n.01}\), being the hypernym of all the synsets shown, has the highest \(\log \det(\Sigma)\) whereas entities that are more specific such as \(\text{object.n.01}, \text{whole.n.02}\) and \(\text{living_thing}\) have decreasingly lower volume.

4.4 Graded Lexical Entailment

HYPERLEX is a lexical entailment dataset which has fine-grained human annotated scores between concept pairs, capturing varying degrees of entailment (Vulic et al., 2016). Concept pairs in HYPERLEX reflect many variants of hypernym relationships, such as no-rel (no lexical relationship), ant (antonyms), syn (synonyms), cohyp (sharing a hypernym but not a hypernym of each other), hyp (hypernym), rhyp (reverse hypernym). We use the noun dataset of HYPERLEX for evaluation, which contains 2,163 pairs.
We evaluate our model by comparing our model scores against the annotated scores. Obtaining a high correlation on a fine-grained annotated dataset is a much harder task compared to a binary prediction since performing well requires meaningful model scores in order to reflect nuances in hypernymy. We use negative divergence as our score for hypernymy scale where large values indicate high degrees of entailment.

We note that the concepts in our trained models are WordNet synsets, where each synset corresponds to a specific meaning of a word. For instance, pop.n.03 has a definition “a sharp explosive sound as from a gunshot or drawing a cork” whereas pop.n.04 corresponds to “music of general appeal to teenagers; ...”. For a given pair of words \((u, v)\), we use the score of the synset pair \((s'_u, s'_v)\) that has the lowest KL divergence among all the pairs \(S_u \times S_v\) where \(S_u, S_v\) are sets of synsets for words \(u\) and \(v\), respectively. More precisely, \(s(u, v) = -\min_{s'_u \in S_u, s'_v \in S_v} D(s'_u, s'_v)\). This pair selection corresponds to choosing the synset pair that has the highest degree of entailment. This approach has been used in word embeddings literature to select most related word pairs (Athiwaratkun & Wilson, 2017). For word pairs that are not in the model, we assign the score equal to the median of all scores.

Table 3 shows that our model using KL divergence and negative sampling approach S1, S2 and S3 outperform all other existing models, achieving state-of-the-art performance for the HyperLex noun dataset. (See Section A.1 for hyperparameter details) Brief summary of other competing models are as follow: FR scores are based on concept word frequency ratio (Weeds et al., 2004). SLQS uses entropy-based measure to quantify entailment (Santus et al.). Vis-ID calculates scores based on visual generality measures (Kiela et al., 2015). WN-B calculates the scores based on the shortest path between concepts in WN taxonomy (Miller, 1995). w2g Guassian embeddings trained using the methodology in Vilnis & McCallum (2014). VOE Vector order embeddings (Vendrov et al., 2015). Euc and Poin calculate scores based on the Euclidean distance and Poincaré distance of the trained Poincaré embeddings (Nickel & Kiela, 2017). The models FR and SLQS are based on word occurrences in text corpus, where FR is trained on the British National Corpus and SLQS is trained on UKWAC, WACKYPEDIA (Bailey & Thompson, 2006; Baroni et al., 2009) and annotated BLESS dataset (Baroni & Lenci, 2011). Other models Vis-ID, w2g, VOE, Euc, Poin and ours are trained on WordNet, with the exception that Vis-ID also uses Google image search results for visual data. The reported results of FR, SLQS, Vis-ID, WN-B, w2g and VOE are from Vulić et al. (2016).
Table 3: Spearman’s correlation for HYPERLEX nouns.

<table>
<thead>
<tr>
<th></th>
<th>FR</th>
<th>SLQS</th>
<th>Vis-ID</th>
<th>WN-B</th>
<th>w2g</th>
<th>VOE</th>
<th>Euc</th>
<th>Poin</th>
<th>HypV</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \rho )</td>
<td>0.283</td>
<td>0.229</td>
<td>0.253</td>
<td>0.240</td>
<td>0.192</td>
<td>0.195</td>
<td>0.389</td>
<td>0.512</td>
<td>0.540</td>
<td>0.590</td>
</tr>
</tbody>
</table>

Table 4: Spearman’s correlation for HYPERLEX nouns for different negative sample schemes.

<table>
<thead>
<tr>
<th>Negative Samples</th>
<th>( \rho )</th>
<th>Negative Samples</th>
<th>( \rho )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 1 \times S_1 )</td>
<td>0.527</td>
<td>( 1 \times S_1 + S_4 )</td>
<td>0.590</td>
</tr>
<tr>
<td>( 2 \times S_1 )</td>
<td>0.529</td>
<td>( 2 \times S_1 + S_4 )</td>
<td>0.580</td>
</tr>
<tr>
<td>( 5 \times S_1 )</td>
<td>0.518</td>
<td>( 5 \times S_1 + S_4 )</td>
<td>0.582</td>
</tr>
<tr>
<td>( 10 \times S_1 )</td>
<td>0.517</td>
<td>( 1 \times S_1 + S_3 )</td>
<td>0.570</td>
</tr>
<tr>
<td>( 1 \times S_1 + S_2 )</td>
<td>0.567</td>
<td>( 2 \times S_1 + S_3 )</td>
<td>0.581</td>
</tr>
<tr>
<td>( 2 \times S_1 + S_2 )</td>
<td>0.567</td>
<td>( S_1 + 0.1 \times S_2 + 0.9 \times S_3 )</td>
<td>0.564</td>
</tr>
<tr>
<td>( 3 \times S_1 + S_2 )</td>
<td>0.584</td>
<td>( S_1 + 0.3 \times S_2 + 0.7 \times S_3 )</td>
<td>0.574</td>
</tr>
<tr>
<td>( 5 \times S_1 + S_2 )</td>
<td>0.561</td>
<td>( S_1 + 0.7 \times S_2 + 0.3 \times S_3 )</td>
<td>0.555</td>
</tr>
<tr>
<td>( 10 \times S_1 + S_2 )</td>
<td>0.550</td>
<td>( S_1 + 0.9 \times S_2 + 0.1 \times S_3 )</td>
<td>0.533</td>
</tr>
</tbody>
</table>

We note that an implementation of Gaussian embeddings model (\( w2g \)) reported by Vulic et al. (2016) does not perform well compared to previous benchmarks such as Vis-ID, FR, SLQS. Our training approach yields the opposite results and outperforms other highly competitive methods such as Poincaré embeddings and Hypervec. This underscores the fact that training approach matters a great deal, even if the concept representation of our work and Vilnis & McCallum (2014)’s are both Gaussian distributions. In addition, we also observe that the vector order embeddings (VOE) do not perform well compared to our model. We hypothesize that it is due to the “soft” orders induced by the divergence penalty that allows our model scores to reflect more closely with hypernymy degrees.

We note another interesting observation that a model trained on a symmetric divergence (ELK) from Section 4.2 can also achieve a high HYPERLEX correlation of 0.532 if KL is used to calculate the model scores. This is because the encapsulation behavior can arise even though the training penalty is symmetric (more explanation in Section 4.2). However, using the symmetric divergence based on ELK results in poor performance on HYPERLEX (0.455), which is expected since it cannot capture the directionality of hypernymy.

We note that another model LEAR obtains an impressive score of 0.686 (Vulic & Mrksic, 2014). However, LEAR use pre-trained word embeddings such as Word2Vec or GloVe as a pre-processing step, leveraging a large vocabulary with rich semantic information. To the best of our knowledge, our model achieves the highest HYPERLEX Spearman’s correlation among models without using large-scale pre-trained embeddings.

Table 4 shows the effects of negative sample section, which can greatly improve HYPERLEX performance. The notation, for instance, \( k \times S_1 + S_2 \) corresponds to using \( k \) samples from \( S_1 \) and 1 sample from \( S_2 \) per each positive sample. We use the best hyperparameters from Section 4.2 for all experiments in Table 4. The setting \( 1 \times S_1 + S_2 + S_4 \) yields the highest HYPERLEX Spearman’s correlation. We observe that the traditional sampling scheme \( S_1 \) alone used by many previous works (Socher et al., 2013; Vendrov et al., 2015) do not yield high scores, and the combination of negative sampling approaches that we proposed help increase the score from the range of 0.53 to 0.56 and above.

5 Future Work

Analogous to recent work by Vulic & Mrksic (2014) which post-processed word embeddings such as GloVe or Word2Vec, our future work includes using the WordNet hierarchy to impose encapsulation orders when training probabilistic embeddings.
In the future, the distribution approach could also be developed for encoder-decoder based models for tasks such as caption generation where the encoder represents the data as a distribution, containing semantic and visual features with uncertainty, and passes this distribution to the decoder which maps to text or images. Such approaches would be reminiscent of variational autoencoders (Kingma & Welling, 2013), which take samples from the encoder’s distribution.

REFERENCES


Enrico Santus, Alessandro Lenci, Qin Lu, and Sabine Schulte im Walde. Chasing hypernyms in vector spaces with entropy. In EACL.


A Supplementary Materials

A.1 Model Hyperparameters

In Section 4.3, the 2-dimensional Gaussian model is trained with S-I method where the number of negative samples is equal to the number of positive samples. The best hyperparameters for \( d = 2 \) model is \((m, \beta, \gamma) = (100.0, 2 \times 10^{-4}, 3.0)\).

In Section 4.2, the best hyperparameters \((m, \beta, \gamma)\) for each of our model are as follows: For Gaussian with KL penalty: \((2000.0, 5 \times 10^{-5}, 500.0)\), Gaussian with reversed KL penalty: \((1000.0, 1 \times 10^{-4}, 1000.0)\), Gaussian with ELK penalty \((1000.0, 1 \times 10^{-5}, 10.0)\).

In Section 4.4, we use the same hyperparameters as in 4.2 with KL penalty, but a different negative sample combination in order to increase the distinguishability of divergence scores. For each positive sample in the training set, we use one sample from each of the methods \(S_1, S_2, S_3\). We note that the model from Section 4.2, using \(S_1\) with the KL penalty obtains a Spearman’s correlation of 0.527.

A.2 Analysis of Training Methodology

We emphasize that Gaussian embeddings have been used in the literature, both in the unsupervised settings where word embeddings are trained with local contexts from text corpus, and in supervised settings where concept embeddings are trained to model annotated data such as WORDNET. The results in supervised settings such as modeling WORDNET have been reported to compare with competing models but often have inferior performance (Vendrov et al., 2015; Vulic et al., 2016). Our paper reaches the opposite conclusion, by showing that a different training approach using Gaussian representations can achieve state-of-the-art results.

A.2.1 Divergence Threshold

Even though the divergence \(D(f||g)\) can capture the extent of encapsulation, a hyponym \(f\) will have the lowest divergence only if \(f = g\). In addition, if \(f\) is a more concentrated distribution that is encompassed by \(g\), \(D(f||g)\) is minimized when \(f\) is at the center of \(g\). However, if there any many hyponyms \(f_1, f_2\) of \(g\), the hyponyms can compete to be close to the center, resulting in too much overlapping between \(f_1\) and \(f_2\) if the random sampling to penalize negative pairs is not sufficiently strong. The divergence threshold \(\gamma\) is used such that there is no longer a penalty once the divergence is below a certain level.

We demonstrate empirically that the threshold \(\gamma\) is important for learning meaningful Gaussian distributions. We fix the hyperparameters \(m = 2000\) and \(\beta = 5 \times 10^{-5}\), with \(S_1\) negative sampling. Figure 5 shows that there is an optimal non-zero threshold and yields the best performance for both WORDNET Hypernym prediction and HYPERLEX Spearman’s correlation. We observe that using \(\gamma = 0\) is detrimental to the performance, especially on HYPERLEX results.

![Figure 5: (a) Spearman’s correlation on HYPERLEX versus \(\gamma\) (b) Test Prediction Accuracy versus \(\gamma\).](a) (b)

A.2.2 Initial Variance Scale

As opposed to the mean vectors that are randomly initialized, we initialize all diagonal covariance elements to be the same. Even though the variance can adapt during training, we find that different initial scales of variance result in drastically different performance. To demonstrate, in Figure 6, we show the best test accuracy and
Table 5: Best results for each loss function for two negative sampling setups: S1 (Left) and S1 + S2 + S4 (Right)

<table>
<thead>
<tr>
<th></th>
<th>Test Accuracy</th>
<th>HYPERLEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eq. 7</td>
<td>0.923</td>
<td>0.527</td>
</tr>
<tr>
<td>Eq. 1</td>
<td>0.886</td>
<td>0.524</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Test Accuracy</th>
<th>HYPERLEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eq. 7</td>
<td>0.911</td>
<td>0.590</td>
</tr>
<tr>
<td>Eq. 1</td>
<td>0.796</td>
<td>0.489</td>
</tr>
</tbody>
</table>

Table 6: Best results for each dimension with negative samples S1 (Left) and S1 + S2 + S4 (Right)

<table>
<thead>
<tr>
<th>d</th>
<th>Test Accuracy</th>
<th>HYPERLEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.909</td>
<td>0.437</td>
</tr>
<tr>
<td>10</td>
<td>0.919</td>
<td>0.462</td>
</tr>
<tr>
<td>20</td>
<td>0.922</td>
<td>0.487</td>
</tr>
<tr>
<td>50</td>
<td>0.923</td>
<td>0.527</td>
</tr>
<tr>
<td>100</td>
<td>0.924</td>
<td>0.526</td>
</tr>
<tr>
<td>200</td>
<td>0.918</td>
<td>0.526</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>d</th>
<th>Test Accuracy</th>
<th>HYPERLEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.901</td>
<td>0.483</td>
</tr>
<tr>
<td>10</td>
<td>0.909</td>
<td>0.526</td>
</tr>
<tr>
<td>20</td>
<td>0.914</td>
<td>0.545</td>
</tr>
<tr>
<td>50</td>
<td>0.911</td>
<td>0.590</td>
</tr>
<tr>
<td>100</td>
<td>0.913</td>
<td>0.573</td>
</tr>
<tr>
<td>200</td>
<td>0.910</td>
<td>0.568</td>
</tr>
</tbody>
</table>

HYPERLEX Spearman’s correlation for each initial variance scale, with other hyperparameters (margin $m$ and threshold $\gamma$) tuned for each variance. We use S1 as a negative sampling method. In general, a low variance scale $\beta$ increases the scale of the loss and requires higher margin $m$ and threshold $\gamma$. We observe that the best prediction accuracy is obtained when $\log(\beta) \approx -10$ or $\beta = 5 \times 10^{-5}$. The best HYPERLEX results are obtained when the scales of $\beta$ are sufficiently low. The intuition is that low $\beta$ increases the scale of divergence $D(\cdot || \cdot)$, which increases the ability to capture relationship nuances.

Figure 6: (a) Spearman’s correlation on HYPERLEX versus $\log(\beta)$ (b) Test Prediction Accuracy versus $\log(\beta)$.

A.2.3 Loss Function

We verify that for this task, our loss function in Equation 7 is superior to Equation 1 originally proposed by Vilnis & McCallum (2014). We use the exact same setup with new negative sample selections and KL divergence thresholding and compare the two loss function. Table 5 verifies our claim.

A.2.4 Dimensionality

Table 6 shows the results for many dimensionalities for two negative sample strategies: S1 and S1 + S2 + S4.

A.2.5 $\alpha$-Divergences

Table 7 show the results using models trained and evaluated with $D(\cdot || \alpha) = D_{\alpha}(\cdot || \alpha)$ with negative sampling approach S1. Interestingly, we found that $\alpha \to 1$ (KL) offers the best result for both prediction accuracy and
HYPERLEX. It is possible that $\alpha = 1$ is sufficiently asymmetric enough to distinguish hypernym directionality, but does not have as sharp penalty as in $\alpha > 1$, which can help learning.

Figure 7: (a) Spearman’s correlation on HYPERLEX versus $\alpha$ (b) Test Prediction Accuracy versus $\alpha$. 