Supplementary Material: Distributional Inclusion Vector Embedding for Unsupervised Hypernymy Detection

Haw-Shiuan Chang[1], **ZiYun Wang**[2], **Luke Vilnis**[1], **Andrew McCallum**[1]
[1]University of Massachusetts, Amherst, USA
[2]Tsinghua University, Beijing, China
hschang@cs.umass.edu, wang-zy14@mails.tsinghua.edu.cn,
{luke, mccallum}@cs.umass.edu

In the supplementary material, we show our experimental details in Section 6, qualitative analysis in Section 7, the experiment for choosing representative scoring functions in Section 8, the performance comparison with previously reported results in Section 9, the experiment of hypernym direction detection in Section 10, the results of DIVE/SBOW trained on PubMed in Section 11, and an efficient way to computing AL_1 scoring function in Section 12.

6 Experimental details

When performing the hypernym detection task, each paper uses different training and testing settings, and we are not aware of a standard setup in this field. For the setting which affects the performance significantly, we try to find possible explanations. For all the settings we tried, we do not find a setting choice which favors a particular embedding/feature space, and all methods use the same training and testing setup in our experiments.

6.1 Training details

We use WaCkypedia corpus (Baroni et al., 2009), a 2009 Wikipedia dump, to compute SBOW and train the embedding. For the datasets without Part of Speech (POS) information (i.e. Medical, LEDS, TM14, Kotlerman 2010, and HyperNet), the training data of SBOW and embeddings are raw text. For other datasets, we concatenate each token with the Part of Speech (POS) of the token before training the models except the case when we need to match the training setup of another paper. All part-of-speech (POS) tags in the experiments come from NLTK.

All words are lower cased. Stop words and rare words (occurs less than 10 times) are removed during our preprocessing step. To train embeddings more efficiently, we chunk the corpus into subsets/lines of 100 tokens instead of using sentence segmentation. Preliminary experiments show that this implementation simplification does not hurt the performance.

BLESS		EVALution		Lenci/Benotto		Weeds		Avg (4 datasets)	
N	OOV	N	OOV	N	OOV	N	OOV	N	OOV
26554	1507	13675	2475	5010	1464	2928	643	48167	6089
Medical		**LEDS**		**TM14**		**Kotlerman 2010**		**HyperNet**	
N	OOV	N	OOV	N	OOV	N	OOV	N	OOV
12602	3711	2770	28	2188	178	2940	89	17670	9424
WordNet		**Avg (10 datasets)**		**HyperLex**					
N	OOV	N	OOV	N	OOV				
8000	3596	94337	24110	2616	59				

Table 5: Dataset sizes. N denotes the number of word pairs in the dataset, and OOV shows how many word pairs are not processed by all the methods in Table 2 and Table 3.

We train DIVE, SBOW, Gaussian embedding, and Word2Vec on only the first 512,000 lines (51.2 million tokens)[1] because we find this way of training setting provides better performances (for both SBOW and DIVE) than training on the whole WaCkypedia or training on randomly sampled 512,000 lines. We suspect this is due to the corpus being sorted by the Wikipedia page titles, which makes some categorical words such as animal and mammal occur 3-4 times more frequently in the first 51.2 million tokens than the rest. The performances of training SBOW PPMI on the whole WaCkypedia is also provided for reference in Table 2 and Table 3. To demonstrate that the quality of DIVE is not very sensitive to the training corpus, we also train DIVE and SBOW PPMI on PubMed and compare the performance of DIVE and SBOW PPMI on Medical dataset in Section 11.

[1]At the beginning, we train the model on this subset just to get the results faster. Later on, we find that in this subset of corpus, the context distribution of the words in testing datasets satisfy the DIH assumption better, so we choose to do all the comparison based on the subset.

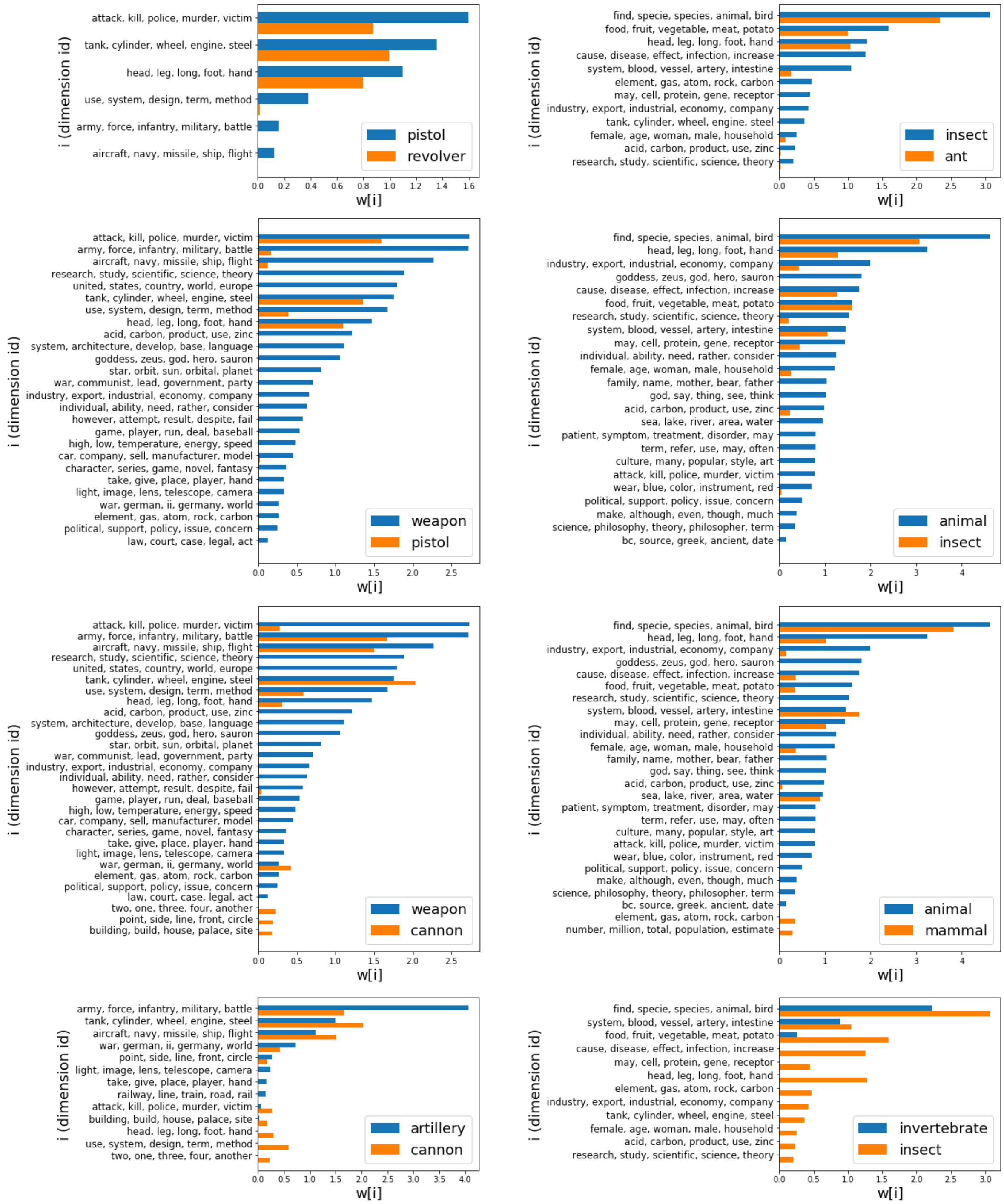

Figure 2: Visualization of the DIVE embedding of word pairs with hypernym relation. The pairs include (revolver,pistol), (pistol,weapon), (cannon,weapon), (artillery,cannon), (ant,insect), (insect,animal), (mammal,animal), and (insect,invertebrate).

6.2 Testing details

The number of testing pairs N and the number of OOV word pairs is presented in Table 5. The micro-average AP is computed by the AP of every dataset weighted by its number of testing pairs N.

In HyperNet and WordNet, some hypernym relations are annotated between phrases instead of words. Phrase embeddings are composed by averaging embeddings (DIVE and skip-grams), or SBOW features of each word. For WordNet, we assume the Part of Speech (POS) tags of the words are the same as the phrase. For Gaussian embed-

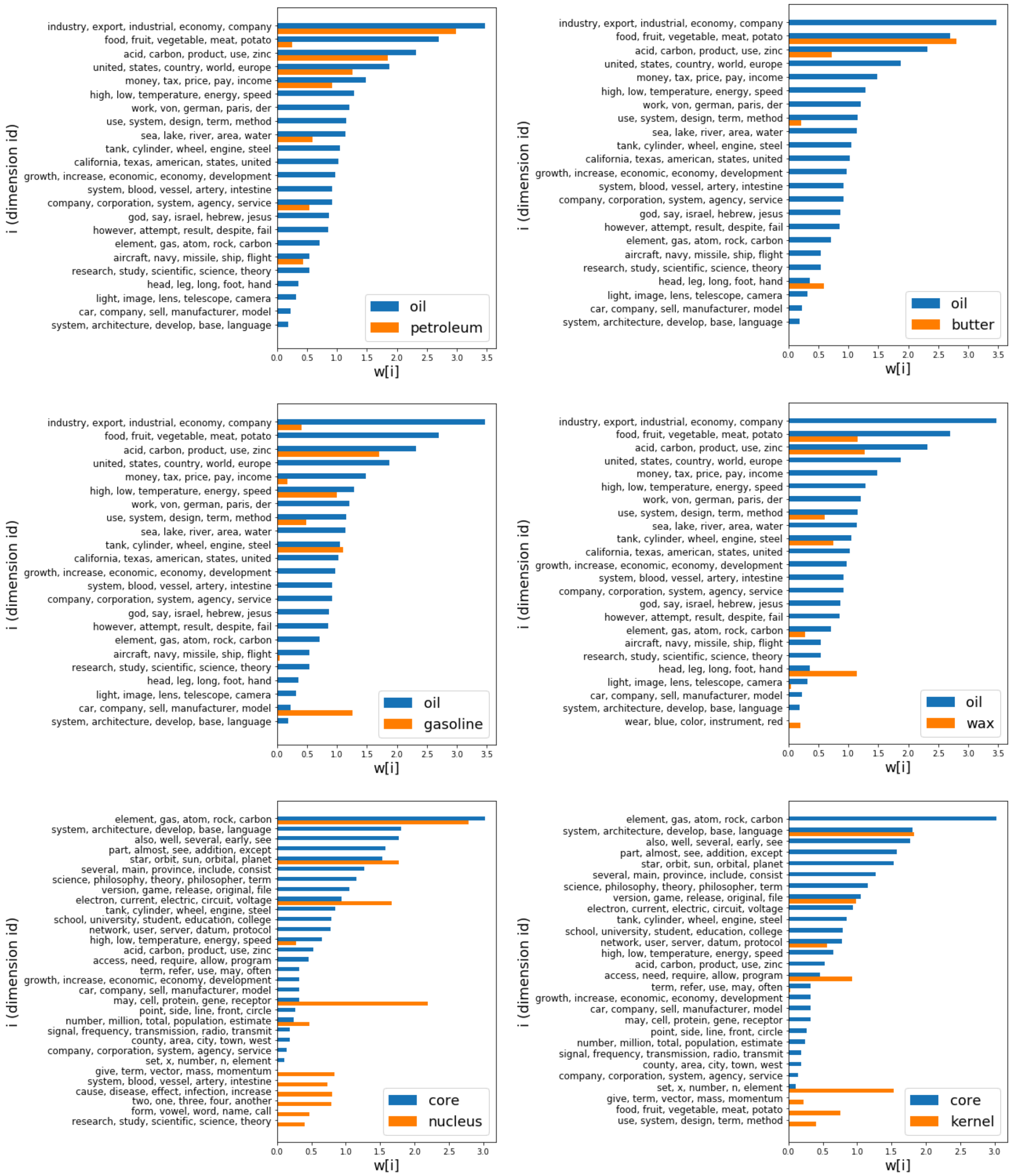

Figure 3: Visualization of the DIVE embedding of oil, core, and their hyponyms.

ding, we use the average score of every pair of words in two phrases when determining the score between two phrases.

6.3 Hyper-parameters

For DIVE, the number of epochs is 15, the learning rate is 0.001, the batch size is 128, the threshold in PMI filter k_f is set to be 30, and the ratio between negative and positive samples (k_I) is 1.5. The hyper-parameters of DIVE were decided based on the performance of HyperNet training set. The window size of skip-grams (Word2Vec) is 10. The number of negative samples (k') in skip-gram is set as 5.

For Gaussian embedding (GE), the number of mixtures is 1, the number of dimensions is 100, the learning rate is 0.01, the lowest variance is 0.1, the highest variance is 100, the highest Gaussian mean is 10, and other hyper-parameters are the default value in https://github.com/benathi/word2gm. The hyper-parameters of GE were also decided based on the performance of HyperNet training set. We also tried to directly tune the hyper-parameters on the micro-average performances of all datasets we are using (except HyperLex), but we found that the performances on most of the datasets are not significantly different from the one tuned by HyperNet.

6.4 Kmeans as NMF

For our K-means (Freq NMF) baseline, K-means hashing creates a $|V| \times 100$ matrix G with orthonormal rows ($G^T G = I$), where $|V|$ is the size of vocabulary, and the (i,k)th element is 0 if the word i does not belong to cluster k. Let the $|V| \times |V|$ context frequency matrix be denoted as M_c, where the (i,j)th element stores the count of word j appearing beside word i. The G created by K-means is also a solution of a type of NMF, where $M_c \approx FG^T$ and G is constrained to be orthonormal (Ding et al., 2005). Hashing context vectors into topic vectors can be written as $M_c G \approx FG^T G = F$.

7 Qualitative analysis

To understand how DIVE preserves DIH more intuitively, we visualize the embedding of several hypernym pairs. In Figure 2, we compare DIVE of different weapons and animals where the dimensions with the embedding value less than 0.1 are removed. We can see that hypernyms often have extra attributes/dimensions that their hyponyms lack. For example, revolver do not appears in the military context as often as pistol do and an ant usually does not cause diseases. We can also tell that cannon and pistol do not have hypernym relation because cannon appears more often in military contexts than pistol.

In DIVE, the signal comes from the count of co-occurring context words. Based on DIH, we can know a terminology to be general only when it appears in diverse contexts many times. In Figure 2, we illustrate the limitation of DIH by showing the DIVE of two relatively rare terminologies: artillery and invertebrate. There are other reasons that could invalid DIH. An example is that a specific term could appear in a special context more often than its hypernym (Shwartz et al., 2017). For instance, gasoline co-occurs with words related to cars more often than oil in Figure 3, and similarly for wax in contexts related to legs or foots. Another typical DIH violation is caused by multiple senses of words. For example, nucleus is the terminology for the core of atoms, cells, comets, and syllables. DIH is satisfied in some senses (e.g. the core of atoms) while not in other senses (the core of cells).

8 Hypernymy scoring functions analysis

Different scoring functions measure different signals in SBOW or embeddings. Since there are so many scoring functions and datasets available in the domain, we introduce and test the performances of various scoring functions so as to select the representative ones for a more comprehensive evaluation of DIVE on the hypernymy detection tasks. We denote the embedding/context vector of the hypernym candidate and the hyponym candidate as $\mathbf{w}_p$ and $\mathbf{w}_q$, respectively.

8.1 Unsupervised scoring functions

Similarity

A hypernym tends to be similar to its hyponym, so we measure the cosine similarity between word vectors of the SBOW features (Levy et al., 2015) or DIVE. We refer to the symmetric scoring function as Cosine or C for short in the following tables. We also train the original skip-grams with 100 dimensions and measure the cosine similarity between the resulting Word2Vec embeddings. This scoring function is referred to as Word2Vec or W.

Generality

The *distributional informativeness hypothesis* (Santus et al., 2014) observes that in many corpora, semantically 'general' words tend to appear more frequently and in more varied contexts. Thus, Santus et al. (2014) advocate using entropy of context distributions to capture the diversity of context. We adopt the two variations of the approach proposed by Shwartz et al. (2017): SLQS Row and SLQS Sub functions. We also refer to SLQS Row as ΔE because it measures the entropy difference of context distributions. For SLQS Sub, the number of top context words is fixed as 100.

Although effective at measuring diversity, the entropy totally ignores the frequency signal from the corpus. To leverage the information, we measure the generality of a word by its L1 norm ($||\mathbf{w}_p||_1$) and L2 norm ($||\mathbf{w}_p||_2$). Recall that Equation 2 indicates that the embedding of the hyper-

Word2Vec (W)	Cosine (C)	SLQS Sub	SLQS Row (ΔE)	Summation (ΔS)	Two norm (ΔQ)
24.8	26.7	27.4	27.6	31.5	31.2
W·ΔE	C·ΔE	W·ΔS	C·ΔS	W·ΔQ	C·ΔQ
28.8	29.5	**31.6**	31.2	31.4	31.1
Weeds	CDE	invCL	Asymmetric L1 (AL_1)		
19.0	31.1	30.7	28.2		

Table 6: Micro average AP@all (%) of 10 datasets using different scoring functions. The feature space is SBOW using word frequency.

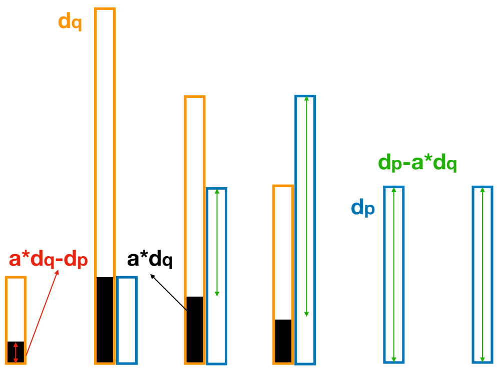

Figure 4: An example of AL_1 distance. If the word pair indeed has the hypernym relation, the context distribution of hyponym ($\mathbf{d}_q$) tends to be included in the context distribution of hypernym ($\mathbf{d}_p$) after proper scaling according to DIH. Thus, the context words only appear beside the hyponym candidate ($a\mathbf{d}_q[c] - \mathbf{d}_p[c]$) causes higher penalty (weighted by w_0).

nym $\mathbf{y}$ should have a larger value at every dimension than the embedding of the hyponym $\mathbf{x}$. When the inclusion property holds, $||\mathbf{y}||_1 = \sum_i \mathbf{y}[i] \geq \sum_i \mathbf{x}[i] = ||\mathbf{x}||_1$ and similarly $||\mathbf{y}||_2 \geq ||\mathbf{x}||_2$. Thus, we propose two scoring functions, difference of vector summation ($||\mathbf{w}_p||_1 - ||\mathbf{w}_q||_1$) and the difference of vector 2-norm ($||\mathbf{w}_p||_2 - ||\mathbf{w}_q||_2$). Notice that when applying the difference of vector summations (denoted as ΔS) to SBOW Freq, it is equivalent to computing the word frequency difference between the hypernym candidate pair.

Similarity plus generality

The combination of 2 similarity functions (Cosine and Word2Vec) and the 3 generality functions (difference of entropy, summation, and 2-norm of vectors) leads to six different scoring functions as shown in Table 6, and C·ΔS is the same scoring function we used in Experiment 1. It should be noted that if we use skip-grams with negative sampling (Word2Vec) as the similarity measurement (i.e., $W \cdot \Delta$ {E,S,Q}), the scores are determined by two embedding/feature spaces together (Word2Vec and DIVE/SBOW).

Inclusion

Several scoring functions are proposed to measure inclusion properties of SBOW based on DIH. Weeds Precision (Weeds and Weir, 2003) and CDE (Clarke, 2009) both measure the magnitude of the intersection between feature vectors ($||\mathbf{w}_p \cap \mathbf{w}_q||_1$). For example, $\mathbf{w}_p \cap \mathbf{w}_q$ is defined by the element-wise minimum in CDE. Then, both scoring functions divide the intersection by the magnitude of the potential hyponym vector ($||\mathbf{w}_q||_1$). invCL (Lenci and Benotto, 2012) (A variant of CDE) is also tested.

We choose these 3 functions because they have been shown to detect hypernymy well in a recent study (Shwartz et al., 2017). However, it is hard to confirm that their good performances come from the inclusion property between context distributions — it is also possible that the context vectors of more general words have higher chance to overlap with all other words due to their high frequency. For instance, considering a one dimension feature which stores only the frequency of words, the naive embedding could still have reasonable performance on the CDE function, but the embedding in fact only memorizes the general words without modeling relations between words (Levy et al., 2015) and loses lots of inclusion signals in the word co-occurrence statistics.

In order to measure the inclusion property without the interference of the word frequency signal from the SBOW or embeddings, we propose a new measurement called asymmetric L_1 distance. We first get context distributions $\mathbf{d}_p$ and $\mathbf{d}_q$ by normalizing $\mathbf{w}_p$ and $\mathbf{w}_q$, respectively. Ideally, the context distribution of the hypernym $\mathbf{d}_p$ will include $\mathbf{d}_q$. This suggests the hypernym distribution $\mathbf{d}_p$ is larger than context distribution of the hyponym with a proper scaling factor $a\mathbf{d}_q$ (i.e., $\max(a\mathbf{d}_q - \mathbf{d}_p, 0)$ should be small). Furthermore, both distributions should be similar, so $a\mathbf{d}_q$ should not be too different from $\mathbf{d}_p$ (i.e., $\max(\mathbf{d}_p - a\mathbf{d}_q, 0)$ should also be small). Therefore, we define asymmetric L1 distance as

$$AL_1 = \min_a \sum_c w_0 \cdot \max(a\mathbf{d}_q[c] - \mathbf{d}_p[c], 0) + \max(\mathbf{d}_p[c] - a\mathbf{d}_q[c], 0), \tag{1}$$

where w_0 is a constant which emphasizes the inclusion penalty. If $w_0 = 1$ and $a = 1$, AL_1 is equivalent to L1 distance. The lower AL_1 distance implies a higher chance of observing the hypernym relation. Figure 4 illustrates a visualization of AL_1 distance. We tried $w_0 = 5$ and $w_0 = 20$. $w_0 = 20$ produces a worse micro-average AP@all on SBOW Freq, SBOW PPMI and DIVE, so we fix w_0 to be 5 in all experiments. An efficient way to solve the optimization in AL_1 is presented in Section 12.

8.2 Results and discussions

We show the micro-average AP@all on 10 datasets using different hypernymy scoring functions in Table 6. We can see the similarity plus generality signals such as C·ΔS and W·ΔS perform the best overall. Among the unnormalized inclusion based scoring functions, CDE works the best. AL_1 performs well compared with other functions which remove the frequency signal such as Word2Vec, Cosine, and SLQS Row. The summation is the most robust generality measurement. In the table, the scoring functions are applied to SBOW Freq, but the performances of hypernymy scoring functions on the other feature spaces (e.g. DIVE) have a similar trend.

9 Comparison with reported results

Each paper uses slightly different setups[2], so it is hard to very fairly compare different approaches. However, by comparing DIVE with reported numbers, we would like to show that the unsupervised methods seem to be previously underestimated, and it is possible for the unsupervised embeddings to achieve performances which are comparable with semi-supervised embeddings when the amount of training data is limited.

9.1 Comparison with SBOW

In Table 7, DIVE with two of the best scoring functions (C·ΔS and W·ΔS) is compared with the previous unsupervised state-of-the-art approaches based on SBOW on different datasets.

There are several reasons which might cause the large performance gaps in some datasets. In addition to the effectiveness of DIVE, some improvements come from our proposed scoring functions. The fact that every paper uses a different training corpus also affects the performances. Furthermore, Shwartz et al. (2017) select the scoring functions and feature space for the first 4 datasets based on AP@100, which we believe is too sensitive to the hyper-parameter settings of different methods.

9.2 Comparison with semi-supervised embeddings

In addition to the unsupervised approach, we also compare DIVE with semi-supervised approaches. When there are sufficient training data, there is no doubt that the semi-supervised embedding approaches such as HyperNet (Shwartz et al., 2016), H-feature detector (Roller and Erk, 2016), and HyperVec (Nguyen et al., 2017) can achieve better performance than all unsupervised methods. However, in many domains such as scientific literature, there are often not many annotated hypernymy pairs (e.g. Medical dataset (Levy et al., 2014)).

Since we are comparing an unsupervised method with semi-supervised methods, it is hard to fairly control the experimental setups and tune the hyper-parameters. In Table 8, we only show several performances which are copied from the original paper when training data are limited[3]. As we can see, the performance from DIVE is roughly comparable to the previous semi-supervised approaches trained on small amount of hypernym pairs. This demonstrates the robustness of our approach and the difficulty of generalizing hypernymy annotations with semi-supervised approaches.

10 Generality estimation and hypernym directionality detection

In Table 9, we show the most general words in DIVE under different queries as constraints. We also present the accuracy of judging which word is

[2]Notice that some papers report F1 instead of AP. When comparing with them, we use 20 fold cross validation to determine prediction thresholds, as done by Roller and Erk (2016).

[3]We neglect the performances from models trained on more than 10,000 hypernym pairs, models trained on the same evaluation datasets with more than 1000 hypernym pairs using cross-validation, and models using other sources of information such as search engines and image classifiers (e.g. the model from Kiela et al. (2015)).

Dataset	BLESS	EVALution	LenciBenotto	Weeds	Medical
Metric	AP@all				F1
Baselines	invCL		APSyn	CDE	Cosine
	5.1	**35.3**	38.2	44.1	23.1
DIVE + C·ΔS	16.3	33.0	50.4	65.5	25.3
DIVE + W·ΔS	**18.6**	32.3	**51.5**	**68.6**	**25.7**
Dataset	LEDS	TM14	Kotlerman 2010	HyperNet	HyperLex
Metric	AP@all			F1	Spearman ρ
Baselines	balAPinc			SLQS	Freq ratio
	73	56	37	22.8	27.9
DIVE + C·ΔS	83.5	57.2	36.6	**41.9**	32.8
DIVE + W·ΔS	**86.4**	**57.3**	**37.4**	38.6	**33.3**

Table 7: Comparison with previous methods based on sparse bag of word (SBOW). All values are percentages. The results of invCL (Lenci and Benotto, 2012), APSyn (Santus et al., 2016), and CDE (Clarke, 2009) are selected because they have the best AP@100 in the first 4 datasets (Shwartz et al., 2017). Cosine similarity (Levy et al., 2015), balAPinc (Kotlerman et al., 2010) in 3 datasets (Turney and Mohammad, 2015), SLQS (Santus et al., 2014) in HyperNet dataset (Shwartz et al., 2016), and Freq ratio (FR) (Vulić et al., 2016) are compared.

Dataset	HyperLex	EVALution	LenciBenotto	Weeds	Medical
Metric	Spearman ρ	AP@all			F1
Baselines	HyperVec (1337)				H-feature (897)
(#Training Hypernymy)	30	**39**	44.8	58.5	**26**
DIVE + C·ΔS (0)	**34.5**	33.8	**52.9**	**70.0**	25.3

Table 8: Comparison with semi-supervised embeddings (with limited training data). All values are percentages. The number in parentheses beside each approach indicates the number of annotated hypernymy word pairs used to train the model. Semi-supervised embeddings include HyperVec (Nguyen et al., 2017) and H-feature (Roller and Erk, 2016). Note that HyperVec ignores POS in the testing data, so we follow the setup when comparing with it.

Query	Top 30 general words					
	use	name	system	include	base	city
	large	state	group	power	death	form
	american	life	may	small	find	body
	design	work	produce	control	great	write
	study	lead	type	people	high	create
species	specie	species	animal	find	plant	may
	human	bird	genus	family	organism	suggest
	gene	tree	name	genetic	study	occur
	fish	disease	live	food	cell	mammal
	evidence	breed	protein	wild	similar	fossil
system	system	use	design	provide	operate	model
	standard	type	computer	application	develop	method
	allow	function	datum	device	control	information
	process	code	via	base	program	software
	network	file	development	service	transport	law

Table 9: We show the top 30 words with the highest embedding magnitude after dot product with the query embedding $\mathbf{q}$ (i.e. showing w such that $||\mathbf{w}^T\mathbf{q}||_1$ is one of the top 30 highest values). The rows with the empty query word sort words based on $||\mathbf{w}||_1$.

a hypernym (more general) given word pairs with hypernym relations in Table 10. The direction is classified correctly if the generality score is greater than 0 (hypernym is indeed predicted as the more general word). For instance, summation difference (ΔS) classifies correctly if $||\mathbf{w}_p||_1 - ||\mathbf{w}_q||_1 > 0$ ($||\mathbf{w}_p||_1 > ||\mathbf{w}_q||_1$).

From the table, we can see that the simple summation difference performs better than SQLS Sub, and DIVE predicts directionality as well as SBOW. Notice that whenever we encounter OOV, the directionality is predicted randomly. If OOV is

Micro Average (10 datasets)	
SBOW Freq + SLQS Sub	SBOW Freq + $\triangle$S
64.4	66.8
SBOW PPMI + $\triangle$S	DIVE + $\triangle$S
66.8	**67.0**

Table 10: Accuracy (%) of hypernym directionality prediction across 10 datasets.

AP@all (%)		Medical				
		CDE	AL_1	ΔS	W·ΔS	C·ΔS
SBOW PPMI	wiki	**23.4**	8.7	13.2	20.1	**24.4**
	PubMed	20.0	7.2	14.2	21.1	23.5
DIVE	wiki	11.7	**9.3**	13.7	**21.4**	19.2
	PubMed	12.6	**9.3**	**15.9**	21.2	20.4

Table 11: Training corpora comparison

excluded, the accuracy of predicting directionality using unsupervised methods can reach around 0.7-0.75.

11 PubMed experiment

To demonstrate that DIVE can compress SBOW in a different training corpus, we train DIVE and SBOW PPMI on biomedical paper abstracts in a subset of PubMed (Wei et al., 2012) and compare their performances on Medical dataset (Levy et al., 2014). We randomly shuffle the order of abstracts, remove the stop words, and only use the first 51.2 million tokens. The same hyper-parameters of DIVE and SBOW PPMI are used, and their AP@all are listed in Table 11. For most scoring functions, the AP@all difference is within 1% compared with the model trained by WaCkypedia.

12 Efficient way to compute asymmetric L1 (AL_1)

Recall that Equation 8 defines AL_1 as follows:

$$\mathcal{L} = \min_a \sum_c w_0 \max(a\mathbf{d}_q[c] - \mathbf{d}_p[c], 0) + \max(\mathbf{d}_p[c] - a\mathbf{d}_q[c], 0),$$

where $\mathbf{d}_p[c]$ is one of dimension in the feature vector of hypernym $\mathbf{d}_p$, $a\mathbf{d}_q$ is the feature vector of hyponym after proper scaling. In Figure 4, an simple example is visualized to illustrate the intuition behind the distance function.

By adding slack variables ζ and ξ, the problem could be converted into a linear programming problem:

$$\begin{aligned}\mathcal{L} = \min_{a,\zeta,\xi}\ & w_0 \sum_c \zeta_c + \sum_c \xi_c \\ & \zeta_c \geq a\mathbf{d}_q[c] - \mathbf{d}_p[c], \quad \zeta_c \geq 0 \\ & \xi_c \geq \mathbf{d}_p[c] - a\mathbf{d}_q[c], \quad \xi_c \geq 0 \\ & a \geq 0,\end{aligned}$$

so it can be simply solved by a general linear programming library.

Nevertheless, the structure of the problem actually allows us to solve this optimization by a simple sorting. In this section, we are going to derive the efficient optimization algorithm.

By introducing Lagrangian multiplier for the constraints, we can rewrite the problem as

$$\begin{aligned}\mathcal{L} = \min_{a,\zeta,\xi}\max_{\alpha,\beta,\gamma,\delta}\ & w_0 \sum_c \zeta_c + \sum_c \xi_c \\ & - \sum_c \alpha_c(\zeta_c - a\mathbf{d}_q[c] + \mathbf{d}_p[c]) \\ & - \sum_c \beta_c(\xi_c - \mathbf{d}_p[c] + a\mathbf{d}_q[c]) \\ & - \sum_c \gamma_c\zeta_c - \sum_c \delta_c\xi_c \\ & \zeta_c \geq 0, \xi_c \geq 0, \alpha_c \geq 0, \beta_c \geq 0, \\ & \gamma_c \geq 0, \delta_c \geq 0, a \geq 0\end{aligned}$$

First, we eliminate the slack variables by taking derivatives with respect to them:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \zeta_c} &= 0 = 1 - \beta_c - \delta_c \\ & \delta_c = 1 - \beta_c, \quad \beta_c \leq 1 \\ \frac{\partial \mathcal{L}}{\partial \xi_c} &= 0 = 1 - \gamma_c - \alpha_c \\ & \gamma_c = w_0 - \alpha_c, \quad \alpha_c \leq w_0.\end{aligned}$$

By substituting in these values for γ_c and δ_c, we get rid of the slack variables and have a new Lagrangian:

$$\begin{aligned}\mathcal{L} = \min_a \max_{\alpha,\beta} & - \sum_c \alpha_c(-a\mathbf{d}_q[c] + \mathbf{d}_p[c]) \\ & - \sum_c \beta_c(-\mathbf{d}_p[c] + a\mathbf{d}_q[c]) \\ & 0 \leq \alpha_c \leq w_0, 0 \leq \beta_c \leq 1, a \geq 0\end{aligned}$$

We can introduce a new dual variable $\lambda_c = \alpha_c -$

$\beta_c + 1$ and rewrite this as:

$$\mathcal{L} = \min_a \max_\lambda \sum_c (\lambda_c - 1)(a\mathbf{d}_q[c] - \mathbf{d}_p[c])$$
$$0 \leq \lambda_c \leq w_0 + 1, a \geq 0$$

Let's remove the constraint on a and replace with a dual variable η:

$$\mathcal{L} = \min_a \max_\lambda \sum_c (\lambda_c - 1)(a\mathbf{d}_q[c] - \mathbf{d}_p[c]) - \eta a$$
$$0 \leq \lambda_c \leq w_0 + 1, \eta \geq 0$$

Now let's differentiate with respect to a to get rid of the primal objective and add a new constraint:

$$\frac{\partial \mathcal{L}}{\partial a} = 0 = \sum_c \lambda_c \mathbf{d}_q[c] - \sum_c \mathbf{d}_q[c] - \eta$$
$$\sum_c \lambda_c \mathbf{d}_q[c] = \sum_c \mathbf{d}_q[c] + \eta$$
$$\mathcal{L} = \max_\lambda \sum_c \mathbf{d}_p[c] - \sum_c \lambda_c \mathbf{d}_p[c]$$
$$\sum_c \lambda_c \mathbf{d}_q[c] = \sum_c \mathbf{d}_q[c] + \eta$$
$$0 \leq \lambda_c \leq w_0 + 1, \eta \geq 0$$

Now we have some constant terms that are just the sums of $\mathbf{d}_p$ and $\mathbf{d}_q$, which will be 1 if they are distributions.

$$\mathcal{L} = \max_\lambda 1 - \sum_c \lambda_c \mathbf{d}_p[c]$$
$$\sum_c \lambda_c \mathbf{d}_q[c] = 1 + \eta$$
$$0 \leq \lambda_c \leq w_0 + 1, \eta \geq 0$$

Now we introduce a new set of variables $\mu_c = \lambda_c \mathbf{d}_q[c]$ and we can rewrite the objective as:

$$\mathcal{L} = \max_\mu 1 - \sum_c \mu_c \frac{\mathbf{d}_p[c]}{\mathbf{d}_q[c]}$$
$$\sum_c \mu_c = 1 + \eta$$
$$0 \leq \mu_c \leq (w_0 + 1)\mathbf{d}_q[c], \eta \geq 0$$

Note that for terms where $\mathbf{d}_q[c] = 0$ we can just set $\mathbf{d}_q[c] = \epsilon$ for some very small epsilon, and in practice, our algorithm will not encounter these because it sorts.

So μ we can think of as some fixed budget that we have to spend up until it adds up to 1, but it has a limit of how much we can spend for each coordinate, given by $(w_0 + 1)\mathbf{d}_q[c]$. Since we're trying to minimize the term involving μ, we want to allocate as much budget as possible to the smallest terms in the summand, and then 0 to the rest once we've spent the budget. This also shows us that our optimal value for the dual variable η is just 0 since we want to minimize the amount of budget we have to allocate.

To make presentation easier, lets assume we sort the vectors in order of increasing $\frac{\mathbf{d}_p[c]}{\mathbf{d}_q[c]}$, so that $\frac{\mathbf{d}_p[1]}{\mathbf{d}_q[1]}$ is the smallest element, etc. We can now give the following algorithm to find the optimal μ.

$$\begin{aligned}
&\text{init } S = 0, c = 1, \mu = 0 \\
&\text{while } S \leq 1: \\
&\quad \mu_c = \min(1 - S, (w_0 + 1)\mathbf{d}_q[c]) \\
&\quad S = S + \mu_c \\
&\quad c = c + 1
\end{aligned}$$

At the end we can just plug in this optimal μ to the objective to get the value of our scoring function.

References

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The WaCky wide web: a collection of very large linguistically processed web-crawled corpora. *Language resources and evaluation* 43(3):209–226.

Daoud Clarke. 2009. Context-theoretic semantics for natural language: an overview. In *workshop on geometrical models of natural language semantics*. pages 112–119.

Chris Ding, Xiaofeng He, and Horst D Simon. 2005. On the equivalence of nonnegative matrix factorization and spectral clustering. In *ICDM*.

Douwe Kiela, Laura Rimell, Ivan Vulic, and Stephen Clark. 2015. Exploiting image generality for lexical entailment detection. In *ACL*.

Lili Kotlerman, Ido Dagan, Idan Szpektor, and Maayan Zhitomirsky-Geffet. 2010. Directional distributional similarity for lexical inference. *Natural Language Engineering* 16(4):359–389.

Alessandro Lenci and Giulia Benotto. 2012. Identifying hypernyms in distributional semantic spaces. In *SemEval*.

Omer Levy, Ido Dagan, and Jacob Goldberger. 2014. Focused entailment graphs for open IE propositions. In *CoNLL*.

Omer Levy, Steffen Remus, Chris Biemann, and Ido Dagan. 2015. Do supervised distributional methods really learn lexical inference relations? In *NAACL-HTL*.

Kim Anh Nguyen, Maximilian Köper, Sabine Schulte im Walde, and Ngoc Thang Vu. 2017. Hierarchical embeddings for hypernymy detection and directionality. In *EMNLP*.

Stephen Roller and Katrin Erk. 2016. Relations such as hypernymy: Identifying and exploiting hearst patterns in distributional vectors for lexical entailment. In *EMNLP*.

Enrico Santus, Tin-Shing Chiu, Qin Lu, Alessandro Lenci, and Chu-Ren Huang. 2016. Unsupervised measure of word similarity: how to outperform co-occurrence and vector cosine in vsms. *arXiv preprint arXiv:1603.09054* .

Enrico Santus, Alessandro Lenci, Qin Lu, and Sabine Schulte Im Walde. 2014. Chasing hypernyms in vector spaces with entropy. In *EACL*.

Vered Shwartz, Yoav Goldberg, and Ido Dagan. 2016. Improving hypernymy detection with an integrated path-based and distributional method. In *ACL*.

Vered Shwartz, Enrico Santus, and Dominik Schlechtweg. 2017. Hypernyms under siege: Linguistically-motivated artillery for hypernymy detection. In *EACL*.

Peter D Turney and Saif M Mohammad. 2015. Experiments with three approaches to recognizing lexical entailment. *Natural Language Engineering* 21(3):437–476.

Ivan Vulić, Daniela Gerz, Douwe Kiela, Felix Hill, and Anna Korhonen. 2016. Hyperlex: A large-scale evaluation of graded lexical entailment. *arXiv preprint arXiv:1608.02117* .

Julie Weeds and David Weir. 2003. A general framework for distributional similarity. In *EMNLP*.

Chih-Hsuan Wei, Bethany R Harris, Donghui Li, Tanya Z Berardini, Eva Huala, Hung-Yu Kao, and Zhiyong Lu. 2012. Accelerating literature curation with text-mining tools: a case study of using pubtator to curate genes in pubmed abstracts. *Database* 2012.