Learning Word Embeddings for Data Sparse and Sentiment Rich Data Sets

Prathusha Kameswara Sarma
Electrical and Computer Engineering
UW - Madison

Abstract

This research proposal describes two algorithms that are aimed at learning word embeddings for data sparse and sentiment rich data sets. The goal is to use word embeddings adapted for domain specific data sets in downstream applications such as sentiment classification. The first approach learns word embeddings in a supervised fashion via SWESA (Supervised Word Embeddings for Sentiment Analysis), an algorithm for sentiment analysis on data sets that are of modest size. SWESA leverages document labels to jointly learn polarity-aware word embeddings and a classifier to classify unseen documents. In the second approach domain adapted (DA) word embeddings are learned by exploiting the specificity of domain specific data sets and the breadth of generic word embeddings. The new embeddings are formed by aligning corresponding word vectors using Canonical Correlation Analysis (CCA) or the related nonlinear Kernel CCA. Experimental results on binary sentiment classification tasks using both approaches for standard data sets are presented.

1 Introduction

Generic word embeddings such as Glove and word2vec (Pennington et al., 2014; Mikolov et al., 2013) which are pre-trained on large sets of raw text, in addition to having desirable structural properties have demonstrated remarkable success when used as features to a supervised learner in various applications such as the sentiment classification of text documents. There are, however, many applications with domain specific vocabularies and relatively small amounts of data. The performance of word embedding approaches in such applications is limited, since word embeddings pre-trained on generic corpora do not capture domain specific semantics/knowledge, while embeddings trained on small data sets are of low quality. Since word embeddings are used to initialize most algorithms for sentiment analysis etc, generic word embeddings further make for poor initialization of algorithms for tasks on domain specific data sets.

A concrete example of a small-sized domain specific corpus is the Substances User Disorders (SUDs) data set (Quanbeck et al., 2014; Litvin et al., 2013), which contains messages from discussion forums for people with substance addictions. These forums are part of mobile health intervention treatments that encourages participants to engage in sobriety-related discussions. The aim with digital intervention treatments is to analyze the daily content of participants' messages and predicit relapse risk. This data is both domain specific and limited in size. Other examples include customer support tickets reporting issues with taxi-cab services, reviews of restaurants and movies, discussions by special interest groups, and political surveys. In general they are common in fields where words have different sentiments from what they would have elsewhere.

Such data sets present significant challenges for algorithms based on word embeddings. First, the data is on specific topics and has a very different distribution from generic corpora, so pre-trained generic word embeddings such as those trained on Common Crawl or Wikipedia are unlikely to yield accurate results in downstream tasks. When performing sentiment classification using pre-trained word embeddings, differences in domains of training and test data sets limit the applicability of the embedding algorithm. For example, in SUDs, discussions are focused on topics related to recovery and addiction; the sentiment behind the word 'party' may be very different in a dating context than in a substance abuse context. Similarly seemingly neutral words such as 'holidays', 'alcohol' etc are indicative of stronger negative sentiment in these domains, while words like 'clean' are indicative of stronger positive sentiment. Thus domain specific vocabularies and word semantics may be a problem for pre-trained sentiment classification models (Blitzer et al., 2007).

Second, there is insufficient data to completely train a word embedding. The SUD data set consists of a few hundred people and only a fraction of these are active (Firth et al., 2017) and (Naslund et al., 2015). This results in a small data set of text messages available for analysis. Furthermore, the content is generated spontaneously on a day to day basis, and language use is informal and unstructured. Running the generic word embedding constructions algorithms on such a data set leads to very noisy outputs that are not suitable as input for downstream applications like sentiment classification. Fine-tuning the generic word embedding also leads to noisy outputs due to the highly nonconvex training objective and the small amount of the data.

This proposal briefly describes two possible solutions to address this problem. Section 3 describes a Canonical Correlation Analysis (CCA) based approach to obtain domain adapted word embeddings. Section 2 describes an biconvex optimization algorithm that jointly learns polarity aware word embeddings and a classifier. Section 4 discusses results from both approaches and outlines potential future work.

2 Supervised Word Embeddings for Sentiment Analysis on Small Sized Data Sets

Supervised Word Embedding for Sentiment Analysis (SWESA) algorithm is an iterative algorithm that minimizes a cost function for both a classifier and word embeddings under unit norm constraint on the word vectors. SWESA incorporates document label information while learning word embeddings from small sized data sets.

2.1 Mathematical model and optimization

Text documents d_i in this framework are represented as a weighted linear combination of words in a given vocabulary. Weights $\boldsymbol{\phi}_i$ used are term frequencies. SWESA aims to find vector representations for words, and by extension of text documents such that applying a nonlinear transformation f to the product $(\theta^\top \mathbf{W}\, \boldsymbol{\phi})$ results in a binary label y indicating the polarity of the document. Mathematically we assume that,

$$\mathbb{P}[Y=1|d=\mathbf{W}\,\boldsymbol{\phi},\boldsymbol{\theta}] = f(\boldsymbol{\theta}^\top \mathbf{W}\,\boldsymbol{\phi}) \qquad (1)$$

for some function f The optimization problem in (1) can be solved as the following minimization problem,

$$\begin{aligned} J(\boldsymbol{\theta},\mathbf{W}) \overset{\text{def}}{=} \frac{-1}{N}\Big[\mathrm{C}_{+}\sum_{y_i=+1}\log\mathbb{P}(Y=y_i|\,\mathbf{W}\,\boldsymbol{\phi}_i,\boldsymbol{\theta}) \\ +\mathrm{C}_{-}\sum_{y_i=-1}\log\mathbb{P}(Y=y_i|\,\mathbf{W}\,\boldsymbol{\phi}_i,\boldsymbol{\theta})\Big] \\ +\lambda_{\boldsymbol{\theta}}||\,\boldsymbol{\theta}\,||_2^2. \end{aligned}$$

This optimization problem can now be written as

$$\begin{aligned} \min_{\substack{\boldsymbol{\theta}\in\mathbb{R}^k,\\ \mathbf{W}\in\mathbb{R}^{k\times V}}} \quad & J(\boldsymbol{\theta},\mathbf{W}) \qquad (2) \\ \text{s.t. } & ||\,\mathbf{w}_j\,||_2 = 1\ \forall j=1,\dots V. \end{aligned}$$

Class imbalance is accounted for by using misclassification costs $\mathrm{C}_{-}, \mathrm{C}_{+}$ as in (Lin et al., 2002). The unit norm constraint in the optimization problem shown in (2) is enforced on word embeddings to discourage degenerate solutions of $\mathbf{w}_j$. This optimization problem is bi-convex. Algorithm 1 shows the algorithm that we use to solve the optimization problem in (2). This algorithm is an alternating minimization procedure that initializes the word embedding matrix $\mathbf{W}$ with $\mathbf{W}_0$ and then alternates between minimizing the objective function w.r.t. the weight vector $\boldsymbol{\theta}$ and the word embeddings $\mathbf{W}$.

The probability model used in this work is logistic regression. Under this assumption the minimization problem in Step 3 of Algorithm 1 is a standard logistic regression problem. In order to solve the optimization problem in line 4 of Algorithm 1 a projected stochastic gradient descent (SGD) with suffix averaging (Rakhlin et al., 2011). Algorithm 2 implements the SGD algorithm (with stochastic gradients instead of full gradients) for solving the optimization problem in step 4 of Algorithm 1. $\mathbf{W}_0$ is initialized via pretrained word2vec embeddings and Latent Semantic Analysis (LSA) (Dumais, 2004) based word

Algorithm 1 Supervised Word Embeddings for Sentiment Analysis (SWESA)

```
Require: $\mathbf{W}_0$, $\Phi$, $C_+$, $C_-$, $\lambda_\theta$, $0 < k < V$, Labels: $\mathbf{y} = [y_1, \ldots, y_N]$, Iterations: $T > 0$,
1: Initialize $\mathbf{W} = \mathbf{W}_0$.
2: for $t = 1, \ldots, T$ do
3:     Solve $\boldsymbol{\theta}_t \leftarrow \arg\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta}, \mathbf{W}_{t-1})$.
4:     Solve $\mathbf{W}_t \leftarrow \arg\min_{\mathbf{W}} J(\boldsymbol{\theta}_t, \mathbf{W})$.
5: end for
6: Return $\boldsymbol{\theta}_T, \mathbf{W}_T$
```

Algorithm 2 Stochastic Gradient Descent for $\mathbf{W}$

```
Require: $\boldsymbol{\theta}, \gamma, \mathbf{W}_0$, Labels: $\mathbf{y} = [y_1, \ldots, y_N]$, Iterations: N, step size: $\eta > 0$, and suffix parameter: $0 < \tau \leq N$.
1: Randomly shuffle the dataset.
2: for $t = 1, \ldots, N$ do
3:     Set $C_t = C_+$ if $y_t = +1$, $C_t = C_-$ if $\underset{\sim}{y_t} = -1$.
4:     $\mathbf{W}_{t+1} = \mathbf{W}_t - \frac{\eta C_t}{1+e^{y_i(\boldsymbol{\theta}^\top \mathbf{W}\,\boldsymbol{\phi}_i)}} \times (-y_i\,\boldsymbol{\theta}\,\boldsymbol{\phi}_i^\top)$
5:     $\mathbf{W}_{t+1,j} = \mathbf{W}_{t+1,j} / || \mathbf{W}_{t+1,j} ||_2 \;\; \forall j = 1, 2, \ldots, V$
6:     $\eta \leftarrow \frac{\eta}{t}$
7: end for
8: Return $\mathbf{W} = \frac{1}{\tau} \sum_{t=N-\tau}^{N} \mathbf{W}_t$
```

embeddings obtained form a matrix of term frequencies from the given data. Dimension k of word vectors is determined empirically by selecting the dimension that provides the best performance across all pairs of training and test data sets.

2.2 Experiment evaluation and results

SWESA is evaluated against the following baselines and data sets,

Datasets: 3 balanced data sets (Kotzias et al., 2015) of 1000 reviews from Amazon, IMDB and Yelp with binary 'positive' and 'negative' sentiment labels are considered. One imbalanced data set with 2500 text messages obtained from a study involving subjects with alcohol addiction is considered. Only 8% of the messages are indicative of 'relapse risk' while the rest are 'benign'. Note that this imbalance influences the performance metrics and can be seen by comparing against the scores achieved by the balanced data sets. Additional information such as number of word tokens etc can be found in the supplemental section.

Data Set	Method	Avg Precision	Avg AUC
Yelp	SWESA (LSA)	78.09±2.84	86.06±2.4
	SWESA (word2vec)	**78.35±4.62**	**86.03**±3.5
	TS (LSA)	76.27±3.0	83.05±5.0
	TS (word2vec)	65.22±4.4	69.08±3.5
	NB	70.31±5.6	57.07±3.3
	RNTN (pre-trained)	83.31±1.1	-
	RNTN (re-trained)	51.15±4.3	-
Amazon	SWESA (LSA)	80.31±3.3	87.54±4.2
	SWESA (word2vec)	**80.36**±2.8	**87.19**±3.3
	TS (LSA)	77.32±4.6	85.00±6.2
	TS (word2vec)	71.09±6.2	77.09±5.3
	NB	72.54±6.4	61.16±4.5
	RNTN (pre-trained)	82.84±0.6	-
	RNTN (re-trained)	49.15±2.1	-
IMDB	SWESA (LSA)	76.40±5.2	81.08±7.6
	SWESA (word2vec)	**77.27±5.4**	**81.04±6.8**
	TS (LSA)	70.36±5.5	77.54±6.8
	TS (word2vec)	56.87±7.6	59.34±8.9
	NB	73.31±5.6	48.40±2.9
	RNTN (pre-trained)	80.88±0.7	-
	RNTN (re-trained)	53.95±1.9	-
A-CHESS	**SWESA (LSA)**	**35.80±2.5**	**83.80±3.1**
	SWESA (word2vec)	35.40±2.0	83.40±2.6
	TS (LSA)	32.20±3.2	**83.80±3.1**
	TS (word2vec)	23.60±2.4	68.00±1.2
	NB	30.30±3.8	45.23±3.3
	RNTN (pre-trained)	-	-
	RNTN (re-trained)	-	-

Table 1: This table shows results from a standard sentiment classification task on all four data sets. Results from SWESA are in boldface and results from pre-trained RNTN are in blue.

- **Naive Bayes:** This is a standard baseline that is best suited for classification in small sized data sets.

- **Recursive Neural Tensor Network:** RNTN is a dependency parser based sentiment analysis algorithm. Both pre-trained RNTN and the RNTN algorithm retrained on the data sets considered here are used to obtain classification accuracy. Note that with the RNTN we do not get probabilities for classes hence we do not compute AUC.

- **Two-Step (TS):** In this set up, embeddings obtained via word2vec on the test data sets and LSA are used to obtain document representation via weighted averaging. Documents are then classified using a Logistic Regressor.

Hyperparameters: Parameters such as dimension of word embeddings, regularization on the logistic regressor etc are determined via 10-fold cross validation.

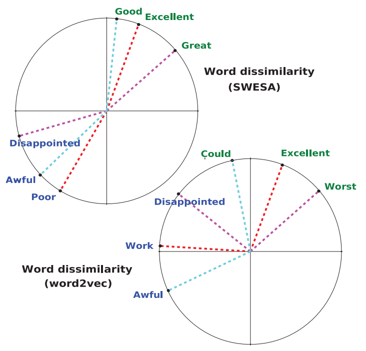

Figure 1: *This figure depicts word embeddings on a unit circle. Cosine angle between embeddings is used to show dissimilar word pairs learned via SWESA and word2vec.*

Results: Average Precision and AUC are reported in table 2. Note that, the word2vec embeddings used in TS are obtained by retraining the word2vec algorithm on the test data sets. To reinforce the point that retraining neural network based algorithms on sparse data sets depreciates their performance, results from pre-trained and retrained RNTN are presented to further support this fact. Since SWESA makes use of document labels when learning word embeddings, resulting word embeddings are polarity aware. Using cosine similarity, word antonym pairs are observed. Given words 'Good,' 'fair' and 'Awful,' the antonym pair 'Good/Awful' is determined via cosine similarity between $\mathbf{w}_{Good}$ and $\mathbf{w}_{Awful}$. Figure 1 shows a small sample of word embeddings learned on the Amazon data set by SWESA and word2vec. The cosine similarity (angle) between the most dissimilar words is calculated and words are depicted as points on the unit circle. These examples illustrate that SWESA captures sentiment polarity at word embedding level despite limited data.

3 Domain Adapted Word Embeddings for Improved Sentiment Classification

While SWESA learns embeddings from domain specific data alone, this approach proposes a method for obtaining high quality Domain Adapted (DA) embeddings by combining generic embeddings and Domain Specific (DS) embeddings via CCA/KCCA. Generic embeddings are trained on large corpora and do not capture domain specific semantics, while DS embeddings are obtained from the domain specific data set via algorithms such as Latent Semantic Analysis (LSA) or other embedding methods. Thus DA embeddings exploit the breath of generic embeddings and the specificity of DS embeddings. The two sets of embeddings are combined using a linear CCA (Hotelling, 1936) or a nonlinear kernel CCA (KCCA) (Hardoon et al., 2004). They are projected along the directions of maximum correlation, and a new (DA) embedding is formed by averaging the projections of the generic embeddings and DS embeddings. The DA embeddings are then evaluated in a sentiment classification setting. Empirically, it is shown that the combined DA embeddings improve substantially over the generic embeddings, DS embeddings and a concatenation-SVD (conc-SVD) based baseline.

3.1 Brief description of CCA/KCCA

Let $\mathbf{W}_{DS} \in \mathbb{R}^{|V_{DS}|\times d_1}$ be the matrix whose columns are the domain specific word embeddings (obtained by, e.g., running the LSA algorithm on the domain specific data set), where V_{DS} is its vocabulary and d_1 is the dimension of the embeddings. Similarly, let $\mathbf{W}_G \in \mathbb{R}^{|V_G|\times d_2}$ be the matrix of generic word embeddings (obtained by, e.g., running the GloVe algorithm on the Common Crawl data), where V_G is the vocabulary and d_2 is the dimension of the embeddings. Let $V_\cap = V_{DS} \cap V_G$. Let $\mathbf{w}_{i,DS}$ be the domain specific embedding of the word $i \in V_\cap$, and $\mathbf{w}_{i,G}$ be its generic embedding. For one dimensional CCA, let ϕ_{DS} and ϕ_G be the projection directions of $\mathbf{w}_{i,DS}$ and $\mathbf{w}_{i,G}$ respectively. Then the projected values are,

$$\begin{aligned}\bar{w}_{i,DS} &= \mathbf{w}_{i,DS}\,\phi_{DS}\\ \bar{w}_{i,G} &= \mathbf{w}_{i,G}\,\phi_G.\end{aligned} \tag{3}$$

CCA maximizes the correlation ρ between $\bar{w}_{i,DS}$ and $\bar{w}_{i,G}$ to obtain ϕ_{DS} and ϕ_G such that

$$\rho(\phi_{DS}, \phi_G) = \max_{\phi_{DS},\phi_G} \frac{\mathbb{E}[\bar{w}_{i,DS}\bar{w}_{i,G}]}{\sqrt{\mathbb{E}[\bar{w}_{i,DS}^2]\mathbb{E}[\bar{w}_{i,G}^2]}} \tag{4}$$

where the expectation is over all words $i \in V_\cap$.

The d-dimensional CCA with $d > 1$ can be defined recursively. Suppose the first $d-1$ pairs of canonical variables are defined. Then the d^{th} pair is defined by seeking vectors maximizing the same correlation function subject to the constraint that they be uncorrelated with the first $d-1$ pairs. Equivalently, matrices of projection vectors $\mathbf{\Phi}_{DS} \in \mathbb{R}^{d_1\times d}$ and $\mathbf{\Phi}_G \in \mathbb{R}^{d_2\times d}$ are obtained for all vectors in $\mathbf{W}_{DS}$ and $\mathbf{W}_G$ where $d \leq$

Data Set		Embedding	Avg Precision	Avg F-score	Avg AUC
Yelp	$\mathbf{W}_{DA}$	KCCA(Glv, LSA)	85.36± 2.8	81.89±2.8	82.57±1.3
		CCA(Glv, LSA)	83.69± 4.7	79.48±2.4	80.33±2.9
		KCCA(w2v, LSA)	87.45± 1.2	83.36±1.2	84.10±0.9
		CCA(w2v, LSA)	84.52± 2.3	80.02±2.6	81.04±2.1
		KCCA(GlvCC, LSA)	**88.11± 3.0**	**85.35±2.7**	**85.80±2.4**
		CCA(GlvCC, LSA)	83.69± 3.5	78.99±4.2	80.03±3.7
		KCCA(w2v, DSw2v)	78.09± 1.7	76.04±1.7	76.66±1.5
		CCA(w2v, DSw2v)	86.22± 3.5	84.35±2.4	84.65±2.2
		concSVD(Glv, LSA)	80.14± 2.6	78.50±3.0	78.92±2.7
		concSVD(w2v, LSA)	85.11± 2.3	83.51±2.2	83.80±2.0
		concSVD(GlvCC, LSA)	84.20± 3.7	80.39±3.7	80.83±3.9
	$\mathbf{W}_{G}$	GloVe	77.13± 4.2	72.32±7.9	74.17±5.0
		GloVe-CC	82.10± 3.5	76.74±3.4	78.17±2.7
		word2vec	82.80± 3.5	78.28±3.5	79.35±3.1
	$\mathbf{W}_{DS}$	LSA	75.36± 5.4	71.17±4.3	72.57±4.3
		word2vec	73.08± 2.2	70.97±2.4	71.76±2.1
Amazon	$\mathbf{W}_{DA}$	KCCA(Glv, LSA)	86.30±1.9	83.00±2.9	83.39±3.2
		CCA(Glv, LSA)	84.68±2.4	82.27±2.2	82.78±1.7
		KCCA(w2v, LSA)	87.09±1.8	82.63±2.6	83.50±2.0
		CCA(w2v, LSA)	84.80±1.5	81.42±1.9	82.12±1.3
		KCCA(GlvCC, LSA)	**89.73±2.4**	85.47±2.4	85.56±2.6
		CCA(GlvCC, LSA)	85.67±2.3	83.83±2.3	84.21±2.1
		KCCA(w2v, DSw2v)	85.68±3.2	81.23±3.2	82.20±2.9
		CCA(w2v, DSw2v)	83.50±3.4	81.31±4.0	81.86±3.7
		concSVD(Glv, LSA)	82.36±2.0	81.30±3.5	81.51±2.5
		concSVD(w2v, LSA)	87.28±2.9	**86.17±2.5**	**86.42±2.0**
		concSVD(GlvCC, LSA)	84.93±1.6	77.81±2.3	79.52±1.7
	$\mathbf{W}_{G}$	GloVe	81.58±2.5	77.62±2.7	78.72±2.7
		GloVe-CC	79.91±2.7	81.63±2.8	81.46±2.6
		word2vec	84.55±1.9	80.52±2.5	81.45±2.0
	$\mathbf{W}_{DS}$	LSA	82.65±4.4	73.92±3.8	76.40±3.2
		word2vec	74.20±5.8	72.49±5.0	73.11±4.8
IMDB	DA	KCCA(Glv, LSA)	73.84±1.3	73.07±3.6	73.17±2.4
		CCA(Glv, LSA)	73.35±2.0	73.00±3.2	73.06±2.0
		KCCA(w2v, LSA)	**82.36±4.4**	**78.95±2.7**	**79.66±2.6**
		CCA(w2v, LSA)	80.66±4.5	75.95±4.5	77.23±3.8
		KCCA(GlvCC, LSA)	54.50±2.5	54.42±2.9	53.91±2.0
		CCA(GlvCC, LSA)	54.08±2.0	53.03±3.5	54.90±2.1
		KCCA(w2v, DSw2v)	60.65±3.5	58.95±3.2	58.95±3.7
		CCA(w2v, DSw2v)	58.47±2.7	57.62±3.0	58.03±3.9
		concSVD(Glv, LSA)	73.25±3.7	74.55±3.2	73.02±4.7
		concSVD(w2v, LSA)	53.87±2.2	51.77±5.8	53.54±1.9
		concSVD(GlvCC, LSA)	78.28±3.2	77.67±3.7	74.55±2.9
	$\mathbf{W}_{G}$	GloVe	64.44±2.6	65.18±3.5	64.62±2.6
		GloVe-CC	50.53±1.8	62.39±3.5	49.96±2.3
		word2vec	78.92±3.7	74.88±3.1	75.60±2.4
	$\mathbf{W}_{DS}$	LSA	67.92±1.7	69.79±5.3	69.71±3.8
		word2vec	56.87±3.6	56.04±3.1	59.53±8.9
A-CHESS	DA	KCCA(Glv, LSA)	32.07±1.3	39.32±2.5	**65.96±1.3**
		CCA(Glv, LSA)	32.70±1.5	35.48±4.2	62.15±2.9
		KCCA(w2v, LSA)	33.45±1.3	**39.81±1.0**	65.92±0.6
		CCA(w2v, LSA)	33.06±3.2	34.02±1.1	60.91±0.9
		KCCA(GlvCC, LSA)	**36.38±1.2**	34.71±4.8	**61.36±2.6**
		CCA(GlvCC, LSA)	32.11±2.9	36.85±4.4	62.99±3.1
		KCCA(w2v, DSw2v)	25.59±1.2	28.27±3.1	57.25±1.7
		CCA(w2v, DSw2v)	24.88±1.4	29.17±3.1	57.76±2.0
		concSVD(Glv, LSA)	27.27±2.9	34.45±3.0	61.59±2.3
		concSVD(w2v, LSA)	29.84±2.3	36.32±3.3	62.94±1.1
		concSVD(GlvCC, LSA)	28.09±1.9	35.06±1.4	62.13±2.6
	$\mathbf{W}_{G}$	GloVe	30.82±2.0	33.67±3.4	60.80±2.3
		GloVe-CC	38.13±0.8	27.45±3.1	57.49±1.2
		word2vec	32.67±2.9	31.72±1.6	59.64±0.5
	$\mathbf{W}_{DS}$	LSA	27.42±1.6	34.38±2.3	61.56±1.9
		word2vec	24.48±0.8	27.97±3.7	57.08±2.5

Table 2: This table shows results from the classification task using sentence embeddings obtained from weighted averaging of word embeddings. Metrics reported are average Precision, F-score and AUC and the corresponding standard deviations (STD). Best results are attained by KCCA (GlvCC, LSA) and are highlighted in boldface.

$\min\{d_1, d_2\}$. Embeddings obtained by $\bar{\mathbf{w}}_{i,DS} = \mathbf{w}_{i,DS}\,\mathbf{\Phi}_{DS}$ and $\bar{\mathbf{w}}_{i,G} = \mathbf{w}_{i,G}\,\mathbf{\Phi}_{G}$ are projections along the directions of maximum correlation.

The final domain adapted embedding for word i is given by $\hat{\mathbf{w}}_{i,DA} = \alpha\bar{\mathbf{w}}_{i,DS} + \beta\bar{\mathbf{w}}_{i,G}$, where the parameters α and β can be obtained by solving the

Data Set	Embedding	Avg Precision	Avg F-score	Avg AUC
Yelp	GlvCC	86.47±1.9	83.51±2.6	83.83±2.2
	KCCA(GlvCC, LSA)	**91.06±0.8**	**88.66±2.4**	**88.76±2.4**
	CCA(GlvCC, LSA)	86.26±1.4	82.61±1.1	83.99±0.8
	concSVD(GlvCC,LSA)	85.53±2.1	84.90±1.7	84.96±1.5
	RNTN	83.11±1.1	-	-
Amazon	GlvCC	87.93±2.7	82.41±3.3	83.24±2.8
	KCCA(GlvCC, LSA)	**90.56±2.1**	**86.52±2.0**	**86.74±1.9**
	CCA(GlvCC, LSA)	87.12±2.6	83.18±2.2	83.78±2.1
	concSVD(GlvCC, LSA)	85.73±1.9	85.19±2.4	85.17±2.6
	RNTN	82.84±0.6	-	-
IMDB	GlvCC	54.02±3.2	53.03±5.2	53.01±2.0
	KCCA(GlvCC, LSA)	59.76±7.3	**53.26±6.1**	56.46±3.4
	CCA(GlvCC, LSA)	53.62±1.6	50.62±5.1	**58.75±3.7**
	concSVD(GlvCC, LSA)	52.75±2.3	53.05±6.0	53.54±2.5
	RNTN	**80.88±0.7**	-	-
A-CHESS	GlvCC	52.21±5.1	**55.26±5.6**	**74.28±3.6**
	KCCA(GlvCC, LSA)	**55.37±5.5**	50.67±5.0	69.89±3.1
	CCA(GlvCC, LSA)	54.34±3.6	48.76±2.9	68.78±2.4
	concSVD(GlvCC, LSA)	40.41±4.2	44.75±5.2	68.13±3.8
	RNTN	-	-	-

Table 3: This table shows results obtained by using sentence embeddings from the InferSent encoder in the sentiment classification task. Metrics reported are average Precision, F-score and AUC along with the corresponding standard deviations (STD). Best results are obtained by KCCA (GlvCC, LSA) and are highlighted in boldface.

following optimization,

$$\min_{\alpha,\beta} \|\bar{\mathbf{w}}_{i,DS} - (\alpha\bar{\mathbf{w}}_{i,DS} + \beta\bar{\mathbf{w}}_{i,G})\|_2^2 + \|\bar{\mathbf{w}}_{i,G} - (\alpha\bar{\mathbf{w}}_{i,DS} + \beta\bar{\mathbf{w}}_{i,G})\|_2^2. \quad (5)$$

Solving (5) gives a weighted combination with $\alpha = \beta = \frac{1}{2}$, i.e., the new vector is equal to the average of the two projections:

$$\hat{\mathbf{w}}_{i,DA} = \frac{1}{2}\bar{\mathbf{w}}_{i,DS} + \frac{1}{2}\bar{\mathbf{w}}_{i,G}. \quad (6)$$

Because of its linear structure, the CCA in (4) may not always capture the best relationships between the two matrices. To account for nonlinearities, a kernel function, which implicitly maps the data into a high dimensional feature space, can be applied. For example, given a vector $\mathbf{w} \in \mathbb{R}^d$, a kernel function K is written in the form of a feature map φ defined by $\varphi : \mathbf{w} = (\mathbf{w}_1, \ldots, \mathbf{w}_d) \mapsto \varphi(\mathbf{w}) = (\varphi_1(\mathbf{w}), \ldots, \varphi_m(\mathbf{w}))(d < m)$ such that given $\mathbf{w}_a$ and $\mathbf{w}_b$

$$K(\mathbf{w}_a, \mathbf{w}_b) = \langle\varphi(\mathbf{w}_a), \varphi(\mathbf{w}_b)\rangle.$$

In kernel CCA, data is first projected onto a high dimensional feature space before performing CCA. In this work the kernel function used is a Gaussian kernel, i.e.,

$$K(\mathbf{w}_a, \mathbf{w}_b) = \exp\Big(-\frac{\|\,\mathbf{w}_a - \mathbf{w}_b\,\|^2}{2\sigma^2}\Big).$$

The implementation of kernel CCA follows the standard algorithm described in several texts such as (Hardoon et al., 2004); see reference for details.

3.2 Experimental evaluation and results

DA embeddings are evaluated in binary sentiment classification tasks on four data sets described in Section 2.2. Document embeddings are obtained via i)a standard framework that expresses documents as weighted combination of their constituent word embeddings and ii) by initializing a state of the art sentence encoding algorithm InferSent (Conneau et al., 2017) with word embeddings to obtain sentence embeddings. Encoded sentences are then classified using a Logistic Regressor.

Word embeddings and baselines:

- **Generic word embeddings:** Generic word embeddings used are GloVe[1] from both Wikipedia and common crawl and the word2vec (Skip-gram) embeddings[2]. These generic embeddings will be denoted as Glv, GlvCC and w2v.

- **DS word embeddings:** DS embeddings are obtained via Latent Semantic Analysis (LSA) and via retraining word2vec on the test data sets using the implementation in gensim[3]. DS embeddings via LSA are denoted by LSA and DS embeddings via word2vec are denoted by DSw2v.

- **concatenation-SVD baseline:** Generic and DS embeddings are concatenated to form a single embeddings matrix. SVD is performed on this matrix and the resulting singular vectors are projected onto the d largest singular values to form resultant word embeddings. These meta-embeddings proposed by (Yin and Schütze, 2016) have demonstrated considerable success in intrinsic tasks such as similarities, analogies etc.

Details about dimensions of the word embeddings and kernel hyperparameter tuning are found in the supplemental material.

Note that InferSent is fine-tuned with a combination of GloVe common crawl embeddings and DA embeddings, and concSVD. Since the data sets at hand do not contain all the tokens required to retrain InferSent, we replace word tokens that are common across our test data sets and InferSent training data with the DA embeddings and concSVD.

3.2.1 Discussion of results

From tables 2 and 3 we see that DA embeddings perform better than concSVD as well as the generic and DS word embeddings, when used in a standard classification task as well as when used to initialize a sentence encoding algorithm. As expected LSA DS embeddings provide better results than word2vec DS embeddings. Also since the A-CHESS dataset is imbalanced, we look at precision closely over the other metric since the positive class is in minority. These results are because i) CCA/KCCA provides an intuitively better technique to preserve information from both the generic and DS embeddings. On the other hand the concSVD based embeddings do not exploit information in both the generic and DS embeddings. ii) Furthermore, in their work (Yin and Schütze, 2016) propose to learn an 'ensemble' of meta-embeddings by learning weights to combine different generic word embeddings via a simple neural network. Via the simple optimization problem we propose in equation (5), we determine the proper weight for combination of DS and generic embeddings in the CCA/KCCA space. Thus, task specific DA embeddings formed by a proper weighted combination of DS and generic word embeddings are expected to do better than the concSVD and other embeddings and this is verified empirically. Also note that the LSA DS embeddings do better than the word2vec DS embeddings. This is expected due to the size of the test sets and the nature of the word2vec algorithm. We expect similar observations when using GloVe DS embeddings owing to the similarities between word2vec and GloVe.

4 Future work and Conclusions

From these initial preliminary results we can see that while SWESA learns embeddings from the domain specific data sets along, DA embeddings combine both generic and domain specific embeddings thereby achieving better performance metrics than SWESA or DS embeddings alone. However, SWESA imparts potentially desirable structural properties to its word embeddings. As a next step we would like to infer from both these approaches to learn better polarized and domain adapted word embeddings.

[1] https://nlp.stanford.edu/projects/glove/

[2] https://code.google.com/archive/p/word2vec/

[3] https://radimrehurek.com/gensim/

References

John Blitzer, Mark Dredze, Fernando Pereira, et al. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*. volume 7, pages 440–447.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364* .

Susan T Dumais. 2004. Latent semantic analysis. *Annual review of information science and technology* 38(1):188–230.

Joseph Firth, John Torous, Jennifer Nicholas, Rebekah Carney, Simon Rosenbaum, and Jerome Sarris. 2017. Can smartphone mental health interventions reduce symptoms of anxiety? a meta-analysis of randomized controlled trials. *Journal of Affective Disorders* .

David R Hardoon, Sandor Szedmak, and John Shawe-Taylor. 2004. Canonical correlation analysis: An overview with application to learning methods. *Neural computation* 16(12):2639–2664.

Harold Hotelling. 1936. Relations between two sets of variates. *Biometrika* 28(3/4):321–377.

Dimitrios Kotzias, Misha Denil, Nando De Freitas, and Padhraic Smyth. 2015. From group to individual labels using deep features. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, pages 597–606.

Yi Lin, Yoonkyung Lee, and Grace Wahba. 2002. Support vector machines for classification in nonstandard situations. *Machine learning* 46(1-3):191–202.

Erika B Litvin, Ana M Abrantes, and Richard A Brown. 2013. Computer and mobile technology-based interventions for substance use disorders: An organizing framework. *Addictive behaviors* 38(3):1747–1756.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. pages 3111–3119.

John A Naslund, Lisa A Marsch, Gregory J McHugo, and Stephen J Bartels. 2015. Emerging mhealth and ehealth interventions for serious mental illness: a review of the literature. *Journal of mental health* 24(5):321–332.

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

Andrew Quanbeck, Ming-Yuan Chih, Andrew Isham, Roberta Johnson, and David Gustafson. 2014. Mobile delivery of treatment for alcohol use disorders: A review of the literature. *Alcohol research: current reviews* 36(1):111.

Alexander Rakhlin, Ohad Shamir, and Karthik Sridharan. 2011. Making gradient descent optimal for strongly convex stochastic optimization. *arXiv preprint arXiv:1109.5647* .

Wenpeng Yin and Hinrich Schütze. 2016. Learning word meta-embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. volume 1, pages 1351–1360.

A Supplemental Material

Dimensions of CCA and KCCA projections. Using both KCCA and CCA, generic embeddings and DS embeddings are projected onto their d largest correlated dimensions. By construction, $d \leq \min(d_1, d_2)$. The best d for each data set is obtained via 10 fold cross validation on the sentiment classification task. Table 2 provides dimensions of all word embeddings considered. Note that for LSA and DA, average word embedding dimension across all four data sets are reported. Generic word embeddings such as GloVe and word2vec are of fixed dimensions across all four data sets.

Kernel parameter estimation. Parameter σ of the Gaussian kernel used in KCCA is obtained empirically from the data. The median (μ) of pairwise distances between data points mapped by the kernel function is used to determine σ. Typically $\sigma = \mu$ or $\sigma = 2\mu$. In this section both values are considered for σ and results with the best performing σ are reported.

Word tokens and word embeddings dimensions:

Table 4 provide the number of unique word tokens in all four data sets used in SWESA as well as in learning DA embeddings.

Data Set	Word Tokens
Yelp	2049
Amazon	1865
IMDB	3075
A-CHESS	3400

Table 4: This table presents the unique tokens present in each of the four data sets considered in the experiments.

Table 5 presents the dimensions of DS and generic word embeddings used to obtain DA embeddings.

Word embedding	Dimension
GloVe	100
word2vec	300
LSA	70
CCA-DA	68
KCCA-DA	68
GloVe common crawl	300
AdaptGloVe	300

Table 5: This table presents the average dimensions of LSA, generic and DA word embeddings.