
Downloaded from [http://umiacs.umd.edu/~jbg/docs/2015_naacl_supervised_anchor.pdf](http://umiacs.umd.edu/~jbg/docs/2015_naacl_supervised_anchor.pdf)

*Contact Jordan Boyd-Graber (jbg@boydgraber.org) for questions about this paper.*
Is Your Anchor Going Up or Down?
Fast and Accurate Supervised Topic Models

Thang Nguyen
iSchool and UMIACS
University of Maryland
and National Library of Medicine,
National Institutes of Health
daithang@umiacs.umd.edu

Jordan Boyd-Graber
Computer Science
University of Colorado Boulder
Jordan.Boyd.Graber@colorado.edu

Jeff Lund,
Kevin Seppi, Eric Ringger
Computer Science
Brigham Young University
{jefflund,kseppi}@byu.edu
ringger@cs.byu.edu

Abstract

Topic models provide insights into document collections, and their supervised extensions also capture associated document-level metadata such as sentiment. However, inferring such models from data is often slow and cannot scale to big data. We build upon the “anchor” method for learning topic models to capture the relationship between metadata and latent topics by extending the vector-space representation of word-cooccurrence to include metadata-specific dimensions. These additional dimensions reveal new anchor words that reflect specific combinations of metadata and topic. We show that these new latent representations predict sentiment as accurately as supervised topic models, and we find these representations more quickly without sacrificing interpretability.

1 Anchors: Speedy Unsupervised Models

The anchor algorithm (Arora et al., 2013) begins with a $V \times V$ matrix $\bar{Q}$ of word co-occurrences, where $V$ is the size of the vocabulary. Each word type defines a vector $\bar{Q}_i$, of length $V$ so that $\bar{Q}_{i,j}$ encodes the conditional probability of seeing word $j$ given that word $i$ has already been seen. Spectral methods (Anandkumar et al., 2012) and the anchor algorithm are fast alternatives to traditional topic model inference schemes because they can discover topics via these summary statistics (quadratic in the number of types) rather than examining the whole dataset (proportional to the much larger number of tokens).

The anchor algorithm takes its name from the idea...
of anchor words—words which unambiguously identify a particular topic. For instance, “wicket” might be an anchor word for the cricket topic. Thus, for any anchor word $a$, $Q_a$. will look like a topic distribution. $Q_{wicket}$ will have high probability for “bowl”, “century”, “pitch”, and “bat”; these words are related to cricket, but they cannot be anchor words because they are also related to other topics.

Because these other non-anchor words could be topically ambiguous, their co-occurrence must be explained through some combination of anchor words; thus for non-anchor word $i$,

$$Q_{i} = \sum_{g_k \in G} C_{i,k} Q_{g_k}$$

where $G = \{g_1, g_2, \ldots, g_K\}$ is the set of $K$ anchor words. The coefficients $C_{i,k}$ of this linear combination correspond to the probability of seeing a topic given a word, from which we can recover the probability of a word given a topic (represented in a matrix $A$) using Bayes’ rule. In our experiments, we follow Arora et al. (2013) to first estimate $Q$ based on the training data and then recover the $C$ matrix

$$C_{i} = \arg\min_{C_i} D_{KL}(Q_{i}, \sum_{g_k \in G} C_{i,k} Q_{g_k})$$

where $D_{KL}(x, y)$ denotes the Kullback-Leibler divergence between $x$ and $y$.

In addition to discovering topics from a given set of anchor words as described above, Arora et al. (2013) also provide a geometric interpretation of a process for finding the needed anchor words. If we view the rows of $Q$ as points in a high-dimensional space, the convex hull of those points provides the anchor words.

Equation 1 linearly combines anchor words’ co-occurrence vectors $Q_{g_k}$ to create the representation of other words. The convex hull corresponds to the perimeter of the space of all possible co-occurrence vectors that can be formed from the set of basis anchor vectors. However, the convex hull only encodes an unsupervised view of the data. To capture topics informed by metadata such as sentiment, we need to explicitly represent the combination of words and metadata.

One problem inherited by the anchor method from parametric topic models is the determination of the number of anchor words (and thus topics) to use. Because word co-occurrence statistics live in an extremely high-dimensional space, the number of anchor words needed to cover all of the data will be quite high. Thus, Arora et al. (2013) require a user to specify the number of anchor words $a priori$ (just as for parametric topic models). They use a form of the Gram-Schmidt process to find the best words that enclose the maximum volume of points.
New column(s) encoding word-sentiment relationship

Figure 2: We form a new column to capture the relationship between words and each sentiment level: per entry is the conditional probability of observing a sentiment level $y^{(l)}$ given an observation of the word $w_i$. Adding all of these columns to $\bar{Q}$ to form an augmented matrix $S$.

2 Supervised Anchor Words

Because the anchor algorithm scales so well compared to traditional probabilistic inference, we now unify the supervised topic models of Section 2 with the anchor algorithm discussed in Section 1. We do so by augmenting the matrix $\bar{Q}$ with an additional dimension for each metadata attribute, such as sentiment. We provide the geometric intuition in Figure 1.

Picture the anchor words projected down to two dimensions (Lee and Mimno, 2014): each word is a point, and the anchor words are the vertices of a polygon encompassing every point. Every non-anchor word can be approximated by a convex combination of the anchor words (Figure 1, top).

Now add an additional dimension as a column to $\bar{Q}$ (Figure 2). This column encodes the metadata specific to a word. For example, we have encoded sentiment metadata in a new dimension (Figure 1, bottom). Neutral sentiment words will stay in the plane inhabited by the other words, positive sentiment words will move up, and negative sentiment words will move down. For simplicity, we only show a single additional dimension, but in general we can add as many dimensions as needed to encode the metadata.

In this new space some of the original anchor words may still be anchor words (“author”). Other words that were near the convex hull boundary in the unaugmented representation may become anchor words in the augmented representation because they capture both topic and sentiment (“anti-lock” vs. “lemon”). Finally, extreme sentiment words might become anchor words in the new higher-dimensional space because they are so important for explaining extreme sentiment values (“wonderful” vs. “awful”).

3.1 Words to Sentiment

Having explained how a word is connected to sentiment, we now elaborate on how to model that connection using the conditional probability of sentiment given a particular word. Assume that sentiment is discretized into a finite set of $L$ sentiment levels $\{y^{(1)}, y^{(2)}, \ldots, y^{(L)}\}$ and that each document is assigned to one of these levels. We define a matrix $S$ of size $V \times (V + L)$. The first $V$ columns are the same as $\bar{Q}$ and the $L$ additional columns capture the relationship of a word to each discrete sentiment
For each additional column $l$, $S_{i, (V+l)} \equiv p(y = y^{(l)} | w = i)$ is the conditional probability of observing a sentiment level $y^{(l)}$ given an observation of word $i$. We compute the conditional probability of a sentiment level $y^{(l)}$ given word $i$

$$S_{i, (V+l)} = \frac{\sum_d (\mathbb{1}[i \in d] \cdot \mathbb{1}[y_d = y^{(l)}])}{\sum_d \mathbb{1}[i \in d]} \quad (2)$$

where the numerator is the number of documents that contain word type $i$ and have sentiment level $y^{(l)}$ and the denominator is the number of documents containing word $i$.

Given this augmented matrix, we again want to find the set of anchor words $\mathcal{G}$ and coefficients $C_{i,k}$ that best capture the relationship between words and sentiment (c.f. Equation 1)

$$S_i = \sum_{g_k \in \mathcal{G}} C_{i,k} S_{g_k} \quad (3)$$

Because we retain the property that non-anchor words are explained through a linear combination of the anchor words, our method retains the same theoretical guarantees of sampling complexity and robustness as the original anchor algorithm.

To facilitate direct comparisons, we keep the number of anchor words fixed in our experiments. Even so, the introduction of metadata forces the anchor method to select the words that best capture this metadata-augmented view of the data. Consequently, some of the original anchor words will remain, and some will be replaced by sentiment-specific anchor words.

4 Quantitative Comparison of Supervised Topic Models

In this section, we evaluate the effectiveness of our new method on a binary sentiment classification problem. Because the supervised anchor algorithm (SUP ANCHOR) finds anchor words (and thus different topics) which capture the sentiment metadata, we evaluate the degree to which its latent representation improves upon the original unsupervised anchor algorithm (Arora et al., 2013, ANCHOR) for classification in terms of both accuracy and speed.

4.1 Sentiment Datasets

We use three common sentiment datasets for evaluation: AMAZON product reviews (Jindal and Liu, 2008), YELP restaurant reviews (Jo and Oh, 2011), and TRIPADVISOR hotel reviews (Wang et al., 2010). For each dataset, we preprocess by tokenizing and removing all non-alphanumeric words and stopwords. As very short reviews are often inscrutable and lack cues to connect to the sentiment, we only consider documents with at least thirty words. We also reduce the vocabulary size by keeping only words that appear in a sufficient number of documents: 50 for AMAZON and YELP datasets, and 150 for TRIPADVISOR (Table 1).

4.2 Documents to Labels

Our goal is to perform binary classification of sentiment. Due to a positive skew of the datasets, the median for all datasets is four out of five. All 5-star reviews are assigned to $y^+$ and the rest of the reviews are assigned to $y^-$. Table 1 summarizes the composition of each dataset and the percentage of documents with high positive sentiment.$^3$

We compare the effectiveness of different representations in predicting high-sentiment documents: unsupervised topic models (LDA), traditional supervised topic models (SLDA), the unmodified anchor algorithm (ANCHOR), our supervised anchor algorithm (SUP ANCHOR), and a traditional TF-IDF (Salton, 1968, TF-IDF) representation of the words.

The anchor algorithm only provides the topic distribution over words; it does not provide the per-document assignment of topics needed to represent the document in a low-dimensional space necessary for producing a prediction $y_d$. Fortunately, this requires a very quick—because the topics are fixed—pass over the documents using a traditional topic model inference algorithm. We use the variational inference implementation for LDA of Blei et al. (2003)$^4$ to obtain $\bar{z}_d$, the topic distribution for document $d$. We consider how the predictive performance on training data varies for each dataset with respect to the number of anchor words fixed in our experiments (Figure 1). The anchor algorithm only provides the topic distribution over words; it does not provide the per-document assignment of topics needed to represent the document in a low-dimensional space necessary for producing a prediction $y_d$. Fortunately, this requires a very quick—because the topics are fixed—pass over the documents using a traditional topic model inference algorithm. We use the variational inference implementation for LDA of Blei et al. (2003)$^4$ to obtain $\bar{z}_d$, the topic distribution for document $d$. For other inference schemes, we use native inference to apply pre-trained topics to extract DEV and TEST topic proportions.
### Table 1: Statistics for the datasets employed in the experiments.

<table>
<thead>
<tr>
<th>Corpus</th>
<th>Train Documents</th>
<th>Test Documents</th>
<th>Tokens</th>
<th>Types</th>
<th>Percentage with Positive Sentiment</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMAZON</td>
<td>13,300</td>
<td>3,314</td>
<td>1,031,659</td>
<td>2,662</td>
<td>52.2%</td>
</tr>
<tr>
<td>TRIP ADVISOR</td>
<td>115,384</td>
<td>3,482</td>
<td>1,142,555</td>
<td>2,958</td>
<td>27.7%</td>
</tr>
<tr>
<td>YELP</td>
<td>13,955</td>
<td>3,482</td>
<td>1,142,555</td>
<td>2,958</td>
<td>27.7%</td>
</tr>
</tbody>
</table>

### Classifiers

Given a low-dimensional representation of a test document, we predict the document’s sentiment $y_{id}$. We have already inferred the topic distribution $z_d$ for each document, and we use $\log(z_d)$ as the features for a classifier. Feature vectors from training data are used to train the classifiers, and feature vectors from the development or test set are used to evaluate the classifiers.

We run three standard machine learning classifiers: decision trees (Quinlan, 1986), logistic regression (Friedman et al., 1998), and a discriminative classifier. For decision trees (hence TREE) and logistic regression (hence LOGISTIC), we use SKLEARN.\(^6\)

For the discriminative classifier, we use a linear classifier with hinge loss (hence HINGE) in Vowpal Wabbit.\(^7\) Because HINGE outputs a regression value in [0, 1], we use a threshold 0.5 to make predictions.

### Parameter Tuning

Parameter tuning is important in topic models, so we cross-validate: each sentiment dataset is split randomly into five folds. We used four folds to form the TRAIN set and reserved the last fold for the TEST set. All cross-validation results are averaged over the four held out DEV sets; the best cross-validation result provides the parameter settings we use on the TEST set.

For ANCHOR and SUP ANCHOR, the parameter for the document-level Dirichlet prior $\alpha$ is required for inferring document-topic distributions given learned topics. Despite selecting this parameter using grid search, $\alpha$ does not affect our final results. The same is also true for SLDA: its predictive performance does not significantly vary as $\alpha$ varies, given a fixed number of topics $K$.\(^8\)

Anchor algorithms are sensitive to the value of anchor threshold $M$ (the minimum document frequency for a word to be considered an anchor word). For each number of topics $K$, we perform a grid search to find the best value of $M$. Figure 3 shows the performance trends.

For LDA, we use the Gibbs sampling implementation in Mallet.\(^9\) For training the model, we run LDA with 5,000 iterations; and for inference (on DEV and TEST) of document topic distribution we iterate 100 times, with lag 5 and 50 burn-in iterations. As Mallet accepts $\sum \alpha_i$ as a parameter, we always initialize $\sum \alpha_i = 1$ and only perform a grid search over different values of $\beta$, the hyper-parameter for Dirichlet prior over the per-topic topic-word distribution, starting from 0.01 and doubling until reaching 0.5.

### 4.3 SUP ANCHOR Outperforms ANCHOR

Learning topics that jointly reflect words and metadata improves subsequent prediction. The results for both SUP ANCHOR and ANCHOR on the TEST set are shown in Figure 4. SUP ANCHOR outperforms ANCHOR on all datasets. This trend holds consistently for LOGISTIC, TREE, and HINGE methods for sentiment prediction. For example, with twenty topics on the AMAZON dataset, SUP ANCHOR gives an accuracy of 0.78, compared to 0.72 for ANCHOR.

\(^6\)http://scikit-learn.org/stable/
\(^7\)http://hunch.net/~vw/
\(^8\)We use the SLDA implementation by Chong Wang: http://www.cs.cmu.edu/~chongw/slda/ to estimate $\alpha$.
\(^9\)http://mallet.cs.umass.edu/topics.php
accuracy of 0.71 in comparison to only 0.62 from \textsc{Anchor}. Similarly, with twenty topics on the \textsc{Yelp} dataset, \textsc{Sup Anchor} has 0.77 accuracy while \textsc{Anchor} has 0.74. Our \textsc{Sup Anchor} model is able to incorporate metadata to learn better representations for predicting sentiment. Moreover, in Section 5 we show that \textsc{Sup Anchor} does not need to sacrifice topic quality to gain predictive power.

\subsection{4.4 \textsc{Sup Anchor} Outperforms \textsc{Slda}}

More surprising is that \textsc{Sup Anchor} also outperforms \textsc{Slda}. Like \textsc{Sup Anchor}, \textsc{Slda} jointly learns topics and their relation to metadata such as sentiment. Figure 4 shows that this trend is consistent on all sentiment datasets. On average, \textsc{Sup Anchor} is 2.2 percent better than \textsc{Slda} on \textsc{Amazon}, and 2.0 percent better on both \textsc{Yelp} and \textsc{TripAdvisor}. Furthermore, \textsc{Sup Anchor} is much faster than \textsc{Slda}.

\textsc{Slda} performs worse than \textsc{Sup Anchor} in part because \textsc{Sup Anchor} is able to jointly find specific lexical terms that improve prediction. Nguyen et al. (2013) show that this improves supervised topic models; forming anchor words around the same strong lexical cues could discover better topics. In contrast, \textsc{Slda} must discover the relationship through the proxy of topics.

\subsection{4.5 Lexical Features}

Ramage et al. (2010) show that interpolating topic and lexical features often provides better classification than either alone. Here, we take the same approach and show how different interpolations of topic and lexical features create better classifiers. We first select an interpolation value $\lambda$ in $\{0, 0.1, 0.2, \ldots, 1\}$, and we then form a new feature vector by concatenating $\lambda$-weighted topic features with $(1 - \lambda)$-weighted lexical features. Figure 5 shows the interplay between topic features and \textsc{TF-IDF} features\textsuperscript{10} as the weight of topic features increases from zero (all \textsc{TF-IDF} features) to one hundred (all \textsc{Sup Anchor} topic features).

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure4}
\caption{Results on \textsc{Test} fold, \textsc{Sup Anchor} outperforms \textsc{Anchor}, \textsc{Lda}, and \textsc{Slda} on all three datasets. We report the results based on \textsc{Logistic} as it produces the best accuracy consistently for \textsc{Anchor}, \textsc{Sup Anchor}, and \textsc{Lda}.}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure5}
\caption{Accuracy on \textsc{Amazon} with twenty topics. \textsc{Sup Anchor} produces good representations for sentiment classification that can be improved by interpolating with lexical \textsc{TF-IDF} features. The interpolation ($x$-axis) ranges from zero (all \textsc{TF-IDF} features) to one hundred (all \textsc{Sup Anchor} topic features).}
\end{figure}

\textsuperscript{10}As before, we do parameter selection on \textsc{Dev} data and report final \textsc{Test} results.

\subsection{4.6 Runtime Analysis}

Having shown that \textsc{Sup Anchor} outperforms both \textsc{Anchor} and \textsc{Slda}, in this section we show that \textsc{Sup Anchor} also inherits the runtime efficiency from \textsc{Anchor}. Table 2 summarizes the runtimes on both \textsc{Amazon} and \textsc{TripAdvisor}; these results were obtained using a six-core 2.8GHz Intel Xeon X5660. On the small dataset \textsc{Amazon}, \textsc{Sup Anchor} finishes the training within one minute, and for the larger \textsc{TripAdvisor} dataset it completes the
Table 2: Runtime statistics (in seconds) for the AMAZON and TRIPADVISOR datasets. Blank cells indicate a timing which does not apply to a particular model. SUP ANCHOR is significantly faster than conventional methods.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Measure</th>
<th>SUP ANCHOR</th>
<th>LDA</th>
<th>SLDA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Preprocessing</td>
<td>32</td>
<td>32</td>
<td>32</td>
</tr>
<tr>
<td>AMAZON</td>
<td>Generating $\bar{Q}/S$</td>
<td>29</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Training</td>
<td>33</td>
<td>886</td>
<td>4,762</td>
</tr>
<tr>
<td></td>
<td>LDAC inference</td>
<td>&lt;5</td>
<td>&lt;5</td>
<td>&lt;5</td>
</tr>
<tr>
<td></td>
<td>Classification</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRIPADVISOR</td>
<td>Preprocessing</td>
<td>305</td>
<td>305</td>
<td>305</td>
</tr>
<tr>
<td></td>
<td>Generating $\bar{Q}/S$</td>
<td>262</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Training</td>
<td>181</td>
<td>8,158</td>
<td>71,967</td>
</tr>
<tr>
<td></td>
<td>LDAC inference</td>
<td>830 (train), 280 (dev/test)</td>
<td>&lt;5</td>
<td>&lt;5</td>
</tr>
<tr>
<td></td>
<td>Classification</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 6: SUP ANCHOR and ANCHOR produce the same topic quality. LDA outperforms all other models and produces the best topics. Performance of SLDA degrades significantly as the number of topic increases.

learning in around three minutes. The main bottleneck for SUP ANCHOR is learning the document distributions over topics, although even this stage is fast for known topic distributions. This result is far better than the twenty hours required by SLDA to train on TRIPADVISOR.

5 Inspecting Anchors and their Topics

One important evaluation for topic models is how easy it is for a human reader to understand the topics. In this section, we evaluate topics produced by each model using topic interpretability (Chang et al., 2009). Topic interpretability measures how human users understand topics presented by a topic modeling algorithm. We use an automated approximation of interpretability that uses a reference corpus as a proxy for which words belong together (Newman et al., 2010). Using half a million documents from Wikipedia, we compute the induced normalized pairwise mutual information (Lau et al., 2014, NPMI) on the top ten words in topics as a proxy for interpretability.

Figure 6 shows the NPMI scores for each model. Unsurprisingly, unsupervised models (LDA) produce the best topic quality. In contrast, supervised models must balance metadata (i.e., response variable) prediction against capturing word meaning. Consequently, SLDA does slightly worse with respect to topic interpretability.

SUP ANCHOR and ANCHOR produce the same topic quality consistently on all datasets. Since SUP ANCHOR and ANCHOR have nearly identical runtime, SUP ANCHOR is better suited for supervised tasks because it improves classification without sacrificing interpretability. It is possible that regularization would improve the interpretability of these topics; Nguyen et al. (2014a) show that adding regularization removes overly frequent words from anchor-discovered topics.

The topics produced by the ANCHOR and SUP ANCHOR algorithms have many similarities. In Table 3, nearly all of the anchor words discovered by ANCHOR are also used by SUP ANCHOR. These anchor words tend to describe general food types, such as
Model Anchor Words and Top Words in Topics

<table>
<thead>
<tr>
<th>Model</th>
<th>Anchor Words and Top Words in Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANCHOR and SUP ANCHOR</td>
<td>pizza burger sushi ice garlic hot amp chicken pork French sandwich coffee cake steak beer fish wine wine restaurant dinner menu nice night bar table meal experience</td>
</tr>
<tr>
<td>ANCHOR</td>
<td>hour wait hour people minutes line long table waiting worth order late night wait people pretty love your friends restaurant open</td>
</tr>
<tr>
<td>SUP ANCHOR</td>
<td>favorite love favorite i ve amazing delicious restaurant eat menu fresh awesome decent pretty didn t restaurant ordered decent wasn t nice night bad stars line line wait people long tacos worth order waiting minutes taco</td>
</tr>
</tbody>
</table>

Table 3: Comparing topics generated for the YELP dataset: anchor words shared by both ANCHOR and SUP ANCHOR are listed. Unique anchor words for each algorithm are listed along with the top ten words for that topic. For clarity, we pruned words which appear in more than 3000 documents as these words appear in every topic. The distinct anchor words reflect positive (“favorite”) and negative (“line”) sentiment rather than less sentiment-specific qualities of restaurants (e.g., restaurants open “late”).

“Pizza” or “burger”, and characterize the YELP dataset well. The similarity of these shared topics explains why both ANCHOR and SUP ANCHOR achieve similar topic interpretability scores.

To explain the predictive power of SUP ANCHOR we must examine the anchor words and topics unique to both algorithms. The anchor words which are unique to ANCHOR include a general topic about wine, and two somewhat coherent topics related to time. By adding supervision to the model we get three new anchor words which identify sentiment ranging from extremely positive reviews mentioning a favorite restaurant to extremely negative reviews complaining about long waits.

This general trend is seen across each of the datasets. For example, ANCHOR and SUP ANCHOR both discover shared topics describing consumer goods, but SUP ANCHOR replaces two topics discussing headphones with topics describing “frustrating” products and “great” products. Similarly, in the TRIPADVISOR data, both ANCHOR and SUP ANCHOR share topics about specific destinations, but only SUP ANCHOR discovers a topic describing “disgusting” hotel rooms.

6 Related Work

Improving the scalability of statistical learning has taken many forms: creating online approximations of large batch algorithms (Hoffman et al., 2013; Zhai et al., 2014) or improving the efficiency of sampling (Yao et al., 2009; Hu and Boyd-Graber, 2012; Li et al., 2014).

These insights have also improved supervised topic models. For example, Zhu et al. (2013) train the max-margin supervised topic models MEDLDA (Zhu et al., 2009) by reformulating the model such that the hinge loss is included inside a collapsed Gibbs sampler, rather than being applied externally on the sampler using costly SVMs. Using insights from Smola and Narayananmurthy (2010), the samplers run in parallel to train the model. While these advancements improve the scalability of max-margin supervised topic models, the improvement is limited by the fact that the sampling algorithm grows with the number of tokens.

In contrast, this paper explores a different vein of research that focuses on using efficient representations of summary statistics to estimate statistical models. While this has seen great success in unsupervised models (Cohen and Collins, 2014), it has increasingly also been applied to supervised models. Wang and Zhu (2014) show how to use tensor decomposition to estimate the parameters of SLDA instead of sampling to find maximum likelihood estimates. In contrast, anchor-based methods rely on non-negative matrix factorization.

We found that a discriminative classifier did not always perform best on the downstream classification task. Zhu et al. (2009) make a comprehensive comparison between MEDLDA, SLDA, and SVM+LDA, and they show that SVM+LDA performs worse than MEDLDA and SLDA on binary classification. It could be that better feature preprocessing could improve our performance.
Bag-of-words representations are not ideal for sentiment tasks. Rubin et al. (2012) introduce Dependency LDA which associates individual word tokens with different labels; their model also outperforms linear SVMs on a very large multi-labeled corpus. Latent variable models that consider grammatical structure (Sayeed et al., 2012; Socher et al., 2011; Iyyer et al., 2014) could also be improved through efficient inference (Cohen and Collins, 2014).

7 Discussion

Supervised anchor word topic modeling provides a general framework for learning better topic representations by taking advantage of both word-cooccurrence and metadata. Our straightforward extension (Equation 2) places each word in a vector space that not only captures co-occurrence with other terms but also the interaction of the word and its sentiment, in contrast to algorithms that only consider raw words.

While our experiments focus on binary classification, the same extension is also applicable to multiclass classification.

Moreover, supervised anchor word topic modeling is fast: it inherits the polynomial-time efficiency from the original unsupervised anchor word algorithm. It is also effective: it is better at providing features for classification than unsupervised topic models and also better than supervised topic models with conventional inference.

Our supervised anchor word algorithm offers the ability to quickly analyze datasets without the overhead of Gibbs sampling or variational inference, allowing users to more quickly understand big data and to make decisions. Combining bag-of-words analysis with metadata through efficient, low-latency topic analysis allows users to have deep insights more quickly.

Acknowledgments We thank Daniel Petersen, Jim Martin, and the anonymous reviewers for their insightful comments. This work was supported by the collaborative NSF Grant IIS-1409287 (UMD) and IIS-1409739 (BYU). Boyd-Graber is also supported by NSF grants IIS-1320538 and NCSE-1422492.

References


Hongning Wang, Yue Lu, and Chengxiang Zhai. 2010. Latent aspect rating analysis on review text data: a rating regression approach. In *Knowledge Discovery and Data Mining*.


