Novel Representations, Regularization & Distances for Text Classification

Konstantinos Skianis
Ph.D. thesis defense
LIX, École Polytechnique, France

Supervisor: Michalis Vazirgiannis
Co-supervisor: Guillaume Wisniewski

March 1, 2019
Abstract

“The complete meaning of a word is always contextual, and no study of meaning apart from context can be taken seriously.”

— John Rupert Firth, 1935

► Fields: natural language processing, machine learning
  ■ help machines perceive text as we do

► Thesis: novel components to fully exploit prior knowledge on text
  ■ better understanding by encoding more information

► Application: text mining
  ■ text = preferred mean of information storage and knowledge transfer
  ■ big data era → scale too large and too time-consuming for humans
Contributions

- Derive new **graph-based text representations** and schemes

- **Linguistic groups** for structured regularization

- Novel approach for **overlapping group regularization**

- Enhance existing **distance techniques** in word embeddings

- Design a **neural architecture** for distance-based classification
Publications (1/2)


(2) Konstantinos Skianis et al. (2016a). “Regularizing Text Categorization with Clusters of Words”. In: EMNLP, pp. 1827–1837


(5) Konstantinos Skianis et al. (2019a). “Boosting Tricks for Word Mover’s Distance”. Manuscript (Submitted in ICWSM 2019)


Not covered in this presentation:


(12) Stamatis Outsios et al. (2018). “Word Embeddings from Large-Scale Greek Web content”. In: *Spoken Language Technology (SLT)*
Outline

1. Introduction
2. Context
3. Graph-based Representations
4. Structured Regularization
5. Sets & Distances
6. Conclusion
Outline

1. Introduction
2. Context
3. Graph-based Representations
4. Structured Regularization
5. Sets & Distances
6. Conclusion
Solving real-world problems
   ▶ data-driven approaches → breakthroughs in research & industry
   ▶ autonomous driving, protein structure prediction, virtual assistants

Why now?
   ▶ strong AI algorithms
   ▶ large hardware power available
   ▶ huge data

What?¹
   ▶ 2.5 quintillion ($10^{18}$) bytes of data created each day
   ▶ Internet of Things (IoT)
   ▶ 90% was generated in the last two years
   ▶ image, video, sequences, text

¹https://www.forbes.com/
Text is everywhere\textsuperscript{2}

\textsuperscript{2}https://www.domo.com/learn/data-never-sleeps-6
Motivation

How can we harvest the full potential of all this textual data?

- Distributional hypothesis (Harris, 1954): words that occur in the same contexts tend to have similar meanings

- "You shall know a word by the company it keeps" (Firth, 1957)

- Text consists of latent concepts corresponding to distributions of observable words

- We need effective and efficient methods to mine them
Outline

1. Introduction
2. Context
3. Graph-based Representations
4. Structured Regularization
5. Sets & Distances
6. Conclusion
Preliminary NLP concepts

- A dataset of data points corresponds to a collection of documents

- A document is a piece of raw text we are interested in as a whole
  - a Web page, a tweet, a user review, a news article, etc.

- A document is a sequence of words: \( d = (t_1, t_2, \ldots, t_{|d|}) \)

- Each word belongs to a common vocabulary

- A term is a processed word, i.e., we apply dimensionality reduction to the vocabulary (e.g., stemming or stopword removal)
## Application

**Text classification (TC)**

Assigning categories to documents (web page, book, media articles etc.)

- TC still one of the most popular tasks
- Spam filtering, email routing, sentiment analysis, qa, chatbots

Basic pipeline of the text classification task.
Humans decide what best means, i.e. provide ground-truth data, a gold standard for the expected outcome

- TC $\rightarrow$ set of documents with golden class labels

We compare a system’s output with the expected outcome:

- **Accuracy**: proportion of good predictions
- macro-average **F1-score**: harmonic mean between precision and recall, averaged over documents or categories

Statistical significance:

- quantify the improvement and decide if we consider it meaningful or simply due to chance
Outline

1. Introduction
2. Context
3. Graph-based Representations
4. Structured Regularization
5. Sets & Distances
6. Conclusion
Traditional steps of TC

Problem: given a document, choose the best classification label

- Feature extraction + supervised learning

- Each data point (document) is represented as a feature vector
  - Bag-of-Words model $\Rightarrow$ binary, frequency
  - TF-IDF (Sparck Jones, 1972)
  - n-gram features (Baeza-Yates and Ribeiro-Neto, 1999)

- A classifier is learnt on a labeled training set of data points:
  - the most frequent category among closest data points, e.g., kNN
  - the category with maximum a posteriori, e.g., Naive Bayes
  - on which side of a separating hyperplane it falls, e.g., SVM
Main approaches

**Bag-of-Words & Linear Classifiers**
- Document is represented as a multiset of its terms
  - fast and effective with simple classifiers
- The term independence assumption:
  - disregarding co-occurrence; keeping only the frequency
- $n$-gram model
  - restrictive in capturing order of terms, huge dimensionality

**Continuous Vectors & Deep Learning**
- Neural (Kim, 2014); (Johnson and Zhang, 2017)
  - Current state-of-the-art results
  - Large pre-trained embeddings needed
- Use the order of words with CNNs (Johnson and Zhang, 2015)
  - Complex architectures with large resources (GPUs)
- Space and time limitations may arise:
  - Computation can be expensive (Joulin et al., 2017)

→ How can we extract more meaningful features?
The yellow vests movement or yellow jackets movement is according to some press a populist, grassroots political movement for economic justice that began in France in November 2018. The movement is motivated by rising fuel prices, high cost of living, and claims that a disproportionate burden of the government’s tax reforms were falling on the working and middle classes, especially in rural and peri-urban areas.
Graph-based approaches

Graph-based Text Mining, NLP and IR
- TextRank (Mihalcea and Tarau, 2004)
- Graph-of-Words (Rousseau and Vazirgiannis, 2013)

Graph-mining for TC
- Frequent subgraphs (Deshpande et al., 2005); (Nikolentzos et al., 2017) $\leftrightarrow$ frequent subgraph mining $\rightarrow$ high complexity
- Random walks, other graph centrality criteria (Hassan et al., 2007); (Malliaros and Skianis, 2015)

Centrality criteria
- Degree($i$) = $\frac{|\mathcal{N}(i)|}{|V|-1}$
- Closeness($i$) = $\frac{|V|-1}{\sum_{j \in V} \text{dist}(i,j)}$, the sum of the length of the shortest paths between the node and all other nodes in the graph
- Pagerank($i$) = $\frac{1-\alpha}{|V|} + \alpha \sum_{(j,i) \in E} \frac{\text{PR}(j)}{\text{out-deg}(j)}$
Why graphs?

- powerful representation
- huge research literature

Bringing graphs to NLP:

- consider info about $n$-grams
  - expressed by paths in the graph
  - keep the dimensionality low (compared to $n$-grams)
- introduce collection-level GoW
- blend document, collection and label GoWs
- integrate word vector similarities as weights in edges

Intersections reveal the derived schemes.
**Document, collection and label GoWs**

$d_1$: A method for the solution of systems of linear equations

![Graph for Document-level GoWs for $d_1$]

**Label: 0**

$G_{d_1} \rightarrow w_1, d_1 \rightarrow w_2, d_1 \rightarrow w_3, d_1 \rightarrow w_4, d_1 \rightarrow w_5, d_1 \rightarrow w_6, d_1 \rightarrow w_7, d_1$

$d_2$: A solution to a linear system

![Graph for Document-level GoWs for $d_2$]

**Label: 1**

$G_{d_2} \rightarrow w_4, d_2 \rightarrow w_5, d_2 \rightarrow w_6, d_2 \rightarrow w_7, d_2$

**Collection-level GoW $G$.**

![Graph for Collection-level GoW $G$.]

**Label GoWs for two classes.**
Proposed weighting schemes

On the collection GoW, the “Inverse Collection Weight” metric:

$$\text{ICW}(t, \mathcal{D}) = \frac{\max_{v \in \mathcal{D}} \text{TW}(v, \mathcal{D})}{\text{TW}(t, \mathcal{D})}$$

Then, the TW-ICW metric becomes:

$$\text{TW-ICW}(t, d) = \text{TW}(t, d) \times \log(\text{ICW}(t, \mathcal{D}))$$

Given the label GoWs, our weighting scheme is a variant of TW-CRC (Shanavas et al., 2016):

$$\text{LW}(t) = \frac{\max(\text{deg}(t, L))}{\max(\text{avg}(\text{deg}(t, L)), \min(\text{deg}(L)))}$$

Last, the TW-ICW-LW metric becomes:

$$\text{TW-ICW-LW}(t, d) = \text{TW}(t, d) \times \log(\text{ICW}(t, \mathcal{D}) \times \text{LW}(t))$$
Edge weighting using word embeddings

Making the most-out-of graphs via word vectors

Use rich word embeddings in order to extract relationships between terms.

- Inject similarities as weights on edges
  - Reward semantically close words in the document GoW (TW)
  - Penalize them in the collection GoW (ICW)

\[
w(t_1, t_2) = 1 - \frac{\text{sim}^{-1}(t_1, t_2)}{\pi}
\]

\[d_1: \text{A method for the solution of systems of linear equations}\]
Datasets & setup

- Linear SVMs with grid search cross-validation for tuning $C$
- Removed stopwords
- No stemming or lowercase transformation, to match Google’s vectors
- Multi-core document and collection graph construction

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Train</th>
<th>Test</th>
<th>Voc</th>
<th>Avg</th>
<th>#w2v</th>
<th>#ICW</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMDB</td>
<td>1,340</td>
<td>660</td>
<td>32,844</td>
<td>343</td>
<td>27,462</td>
<td>352K</td>
</tr>
<tr>
<td>WebKB</td>
<td>2,803</td>
<td>1,396</td>
<td>23,206</td>
<td>179</td>
<td>20,990</td>
<td>273K</td>
</tr>
<tr>
<td>20NG</td>
<td>11,293</td>
<td>7,528</td>
<td>62,752</td>
<td>155</td>
<td>54,892</td>
<td>1.7M</td>
</tr>
<tr>
<td>Amazon</td>
<td>5,359</td>
<td>2,640</td>
<td>19,980</td>
<td>65</td>
<td>19,646</td>
<td>274K</td>
</tr>
<tr>
<td>Reuters</td>
<td>5,485</td>
<td>2,189</td>
<td>11,965</td>
<td>66</td>
<td>9,218</td>
<td>163K</td>
</tr>
<tr>
<td>Subj.</td>
<td>6,694</td>
<td>3,293</td>
<td>8,639</td>
<td>11</td>
<td>8,097</td>
<td>58K</td>
</tr>
</tbody>
</table>

#ICW: number of edges in the collection-level graph; #w2v: number of words in pre-trained vectors.
## Results (1/2)

<table>
<thead>
<tr>
<th>Methods</th>
<th>20NG (max)</th>
<th>IMDB (sum)</th>
<th>Subjectivity (max)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$w = 3$</td>
<td>$w = 4$</td>
<td>$w = 2$</td>
</tr>
<tr>
<td>F1 Acc</td>
<td>F1 Acc</td>
<td>F1 Acc</td>
<td>F1 Acc</td>
</tr>
<tr>
<td>TF</td>
<td>80.88 81.55</td>
<td>-</td>
<td>84.23 84.24</td>
</tr>
<tr>
<td>w2v</td>
<td>74.43 75.75</td>
<td>-</td>
<td>82.57 82.57</td>
</tr>
<tr>
<td>TF-binary (ngrams)</td>
<td>81.64 82.11*</td>
<td>-</td>
<td>83.02 83.03</td>
</tr>
<tr>
<td>TW (degree)</td>
<td>82.37 83.00*</td>
<td>82.21 82.83*</td>
<td>84.82 84.84 84.67 84.69</td>
</tr>
<tr>
<td>TW (w2v)</td>
<td>81.88 82.51*</td>
<td>82.21 82.87*</td>
<td>84.66 84.69 84.52 84.54</td>
</tr>
<tr>
<td>TF-IDF</td>
<td>82.44 83.01*</td>
<td>-</td>
<td>83.33 83.33</td>
</tr>
<tr>
<td>TF-IDF-w2v</td>
<td>82.52 83.09*</td>
<td>-</td>
<td>82.87 82.87</td>
</tr>
<tr>
<td>TW-IDF (degree)</td>
<td>84.75 85.47*</td>
<td>84.80 85.46*</td>
<td>82.86 82.87 83.02 83.03</td>
</tr>
<tr>
<td>TW-IDF (w2v)</td>
<td>84.66 85.32</td>
<td>84.46 85.13</td>
<td>83.47 83.48 83.31 83.33</td>
</tr>
<tr>
<td>TW-ICW (deg, deg)</td>
<td>85.24 85.80*</td>
<td>85.41 86.05*</td>
<td>84.98 85.00 85.13 85.15</td>
</tr>
<tr>
<td>TW-ICW (w2v)</td>
<td><strong>85.33 85.93</strong>*</td>
<td>85.29 85.90*</td>
<td>85.12 85.15 84.82 84.84</td>
</tr>
<tr>
<td>TW-ICW-LW (deg)</td>
<td>85.01 85.66*</td>
<td>85.02 85.66*</td>
<td>85.73 85.75 85.28 85.30</td>
</tr>
<tr>
<td>TW-ICW-LW (w2v)</td>
<td>82.56 83.11*</td>
<td>82.24 82.81*</td>
<td>85.29 85.30 84.39 84.39</td>
</tr>
<tr>
<td>TW-ICW-LW (pgr)</td>
<td>83.92 84.66</td>
<td>83.80 84.54</td>
<td>84.97 85.00 85.73 85.75</td>
</tr>
<tr>
<td>TW-ICW-LW (cl)</td>
<td>84.61 85.22</td>
<td>84.71 85.27</td>
<td><strong>87.27 87.27</strong></td>
</tr>
</tbody>
</table>

Macro-F1 and accuracy for window size $w$. Bold for best performance on each window size and blue for best overall on a dataset. * indicates statistical significance of improvement over TF at $p < 0.05$ using micro sign test.
## Results (2/2)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Amazon (max)</th>
<th>WebKB (sum)</th>
<th>Reuters (max)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$w = 2$</td>
<td>$w = 3$</td>
<td>$w = 2$</td>
</tr>
<tr>
<td></td>
<td>F1</td>
<td>Acc</td>
<td>F1</td>
</tr>
<tr>
<td>TF</td>
<td>80.68</td>
<td>80.68</td>
<td>-</td>
</tr>
<tr>
<td>w2v</td>
<td>79.05</td>
<td>79.05</td>
<td>-</td>
</tr>
<tr>
<td>TF-binary (ngrams)</td>
<td>79.84</td>
<td>79.84</td>
<td>-</td>
</tr>
<tr>
<td>TW (degree)</td>
<td>80.07</td>
<td>80.07</td>
<td>80.41</td>
</tr>
<tr>
<td>TW (w2v)</td>
<td>80.07</td>
<td>80.07</td>
<td>79.54</td>
</tr>
<tr>
<td>TF-IDF</td>
<td>80.26</td>
<td>80.26</td>
<td>-</td>
</tr>
<tr>
<td>TF-IDF-w2v</td>
<td>80.49</td>
<td>80.49</td>
<td>-</td>
</tr>
<tr>
<td>TW-IDF (degree)</td>
<td>81.47</td>
<td>81.47*</td>
<td>81.55</td>
</tr>
<tr>
<td>TW-IDF (w2v)</td>
<td>79.61</td>
<td>79.62</td>
<td>77.60</td>
</tr>
<tr>
<td>TW-ICW (deg, deg)</td>
<td>82.08</td>
<td>82.08*</td>
<td>82.02</td>
</tr>
<tr>
<td>TW-ICW (w2v)</td>
<td>80.86</td>
<td>80.87*</td>
<td>78.82</td>
</tr>
<tr>
<td>TW-ICW-LW (deg)</td>
<td>82.72</td>
<td>82.72*</td>
<td>82.91</td>
</tr>
<tr>
<td>TW-ICW-LW (w2v)</td>
<td>80.56</td>
<td>80.56</td>
<td>78.32</td>
</tr>
<tr>
<td>TW-ICW-LW (pgr)</td>
<td>82.23</td>
<td>82.23*</td>
<td>82.46</td>
</tr>
<tr>
<td>TW-ICW-LW (cl)</td>
<td><strong>82.90</strong></td>
<td><strong>82.91</strong>*</td>
<td><strong>83.02</strong></td>
</tr>
</tbody>
</table>

Macro-F1 and accuracy for window size $w$. Bold for best performance on each window size and blue for best overall on a dataset. * indicates statistical significance of improvement over TF at $p < 0.05$ using micro sign test.
### Comparison vs state-of-the-art methods

<table>
<thead>
<tr>
<th>Method</th>
<th>20NG</th>
<th>IMDB</th>
<th>SUBJ.</th>
<th>AMAZON</th>
<th>WEBKB</th>
<th>REUTERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN (no w2v, 20 ep.) (Kim, 2014)</td>
<td>83.19</td>
<td>74.09</td>
<td>88.16</td>
<td>80.68</td>
<td>88.17</td>
<td>94.75</td>
</tr>
<tr>
<td>CNN (w2v, 20 ep.)</td>
<td>81.92</td>
<td>64.09</td>
<td>89.04</td>
<td>82.08</td>
<td>84.05</td>
<td>95.80</td>
</tr>
<tr>
<td>FastText (100 ep.) (Joulin et al., 2017)</td>
<td>79.70</td>
<td>84.70</td>
<td>88.60</td>
<td>79.50</td>
<td>92.60</td>
<td>97.00</td>
</tr>
<tr>
<td>FastText (w2v, 5 ep.)</td>
<td>80.80</td>
<td>86.10</td>
<td>88.50</td>
<td>80.90</td>
<td>91.40</td>
<td>97.40</td>
</tr>
<tr>
<td>TextRank (Mihalcea and Tarau, 2004)</td>
<td>82.56</td>
<td>83.33</td>
<td>84.78</td>
<td>80.49</td>
<td>92.27</td>
<td>97.35</td>
</tr>
<tr>
<td>Word Attraction (Wang et al., 2015)</td>
<td>61.24</td>
<td>70.75</td>
<td>86.60</td>
<td>78.29</td>
<td>79.46</td>
<td>91.34</td>
</tr>
<tr>
<td>TW-CRC (Shanavas et al., 2016)</td>
<td>85.35</td>
<td>85.15</td>
<td>89.28</td>
<td>81.13</td>
<td>92.71</td>
<td>97.39</td>
</tr>
<tr>
<td>TW-ICW-LW (ours)</td>
<td><strong>86.05</strong></td>
<td><strong>87.27</strong></td>
<td><strong>90.28</strong></td>
<td><strong>83.03</strong></td>
<td><strong>93.57</strong></td>
<td><strong>97.53</strong></td>
</tr>
</tbody>
</table>

Comparison in accuracy(%) to deep learning and graph-based approaches.

### Discussion

- With label graphs used, **word vectors** do not improve accuracy
  - terms concerning different labels can be close in vector space
- Closeness in document GoW → best performance in 3/6
  - can only have an affect in larger document lengths and when used along with label graphs
Examining the window size

F1 score (left) and accuracy (right) of TW, TW-ICW and TW-ICW-LW (all degree) on Reuters, WebKB and Subjectivity, for $w = \{2, \ldots, 10\}$.
Summary

Contributions

▶ a full graph-based framework for TC
▶ determine the importance of a term using node centrality criteria
  ■ document, collection and label level schemes
▶ add word-embedding information as weights on the edges

Future Directions

▶ could also be applied in IR, keyword extraction, summarization etc.
▶ Graph-of-Documents
  ■ Graph comparison via graph kernels (Borgwardt et al., 2007)
  ■ Word Mover’s Distance (Kusner et al., 2015)
▶ Neural Message Passing (Gilmer et al., 2017) & Text GCN (Yao et al., 2019)

What more?
Use these GoW representations for regularization!
Outline

1. Introduction
2. Context
3. Graph-based Representations
4. Structured Regularization
5. Sets & Distances
6. Conclusion
Prediction as loss minimization

Given a training set of $N$ data points $\{(x^i, y^i)\}_{i=1}^N$, find the optimal set of feature weights $\theta^*$ such that:

$$
\theta^* = \arg\min_{\theta} \sum_{i=1}^{N} \mathcal{L}(x^i, \theta, y^i) + \lambda \Omega(\theta)
$$

- **Empirical risk**
- **Penalty term**

A logistic regression loss function:

$$
\mathcal{L}(x, \theta, y) = \log(1 + \exp(-y \theta^T x))
$$

**Regularization**

- huge dimensionality in text
- address overfitting
- more sparse models
- use prior knowledge we may have on the features
Regularization

**L1 and L2 regularization** (Tibshirani, 1996); (Hoerl and Kennard, 1970)

\[
\theta^* = \arg\min_{\theta} \sum_{i=1}^{N} \mathcal{L}(x^i, \theta, y^i) + \lambda \sum_{j=1}^{p} |\theta_j| \text{ (lasso)} \tag{2}
\]

\[
\theta^* = \arg\min_{\theta} \sum_{i=1}^{N} \mathcal{L}(x^i, \theta, y^i) + \lambda \sum_{j=1}^{p} \theta_j^2 \text{ (ridge)} \tag{3}
\]

**Group lasso** (Bakin, 1999); (Yuan and Lin, 2006)

\[
\Omega(\theta) = \lambda \sum_{g} \lambda_g \|\theta_g\|_2 \tag{4}
\]

where \(\theta_g\) is the subset of feature weights restricted to group \(g\).

**Linguistic structured regularizers**

- Sentence, parse, Brown, LDA (Yogatama and Smith, 2014)
Group lasso variants

Grey boxes depict active features. While group lasso selects a whole group, sparse group lasso can select some group’s features. In the overlapping case, groups can share features, while in the last, $L_1$ is applied inside each group.
ADMM for sparse overlapping group lasso

- The objective (Yogatama and Smith, 2014):

\[
\min_{\theta, v} \Omega_{\text{las}}(\theta) + \Omega_{\text{glas}}(v) + \mathcal{L}(\theta)
\]

\[
\text{s.t. } v = M\theta
\]

where \(v\) is a copy-vector of \(\theta\), \(M\) is an indicator matrix of size \(L \times V\), linking \(\theta\) and their copies \(v\).

- An augmented Lagrangian problem is formed:

\[
\Omega_{\text{las}}(\theta) + \Omega_{\text{glas}}(v) + \mathcal{L}(\theta) + u^\top (v - M\theta) + \frac{\rho}{2} \|v - M\theta\|^2
\]

- Essentially, the problem becomes the iterative update of \(\theta\), \(v\) and \(u\):

\[
\min_\theta \Omega_{\text{las}}(\theta) + \mathcal{L}(\theta) + u^\top M\theta + \frac{\rho}{2} \|v - M\theta\|^2
\]

\[
\min_v \Omega_{\text{glas}}(v) + u^\top v + \frac{\rho}{2} \|v - M\theta\|^2
\]

\[
u = u + \rho(v - M\theta)
\]
Contribution

Regularizing with Clusters of Words

- LSI topic modeling ($K$ topics)

\[ \Omega_{LSI}(\theta) = \sum_{k=1}^{K} \lambda \| \theta_k \|_2 \]  

- Community detection on Graph-of-Words ($C$ communities):

\[ \Omega_{gow}(\theta) = \sum_{c=1}^{C} \lambda \| \theta_c \|_2 \]  

- Clustering in word embeddings ($K$ clusters):

\[ \Omega_{word2vec}(\theta) = \sum_{k=1}^{K} \lambda \| \theta_k \|_2 \]
Group lasso pros & cons

**Advantages**
- powerful regularization method
- in general fast
- sparsity

**Drawbacks**
- these groupings are either not available or hard to be extracted
- no ground truth groups of words exist to validate their quality
- group lasso may fail to create sparse models
Orthogonal Matching Pursuit for TC

OMP & Group OMP

Greedy feature selection algorithms used in signal processing. (Mallat and Zhang, 1993)

Based on:

- Group OMP for Variable Selection (Swirszcz et al., 2009)
- Logistic Regression (Lozano et al., 2011)

Contributions:

- use OMP as regularizer
- introduce overlapping Group OMP
Pipeline

Text classification with the OMP regularizer.
Datasets

**Topic categorization**
- 20 NG: four binary classification tasks

**Sentiment analysis**
- movie reviews (Pang and Lee, 2004)
- floor speeches by U.S. Congressmen deciding “yea”/“nay” votes on the bill under discussion (Thomas et al., 2006)
- Amazon product reviews (Blitzer et al., 2007)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Train</th>
<th>Dev</th>
<th>Test</th>
<th>Voc</th>
<th>#Sents</th>
</tr>
</thead>
<tbody>
<tr>
<td>20NG</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>science</td>
<td>949</td>
<td>238</td>
<td>790</td>
<td>25787</td>
<td>16411</td>
</tr>
<tr>
<td>sports</td>
<td>957</td>
<td>240</td>
<td>796</td>
<td>21938</td>
<td>14997</td>
</tr>
<tr>
<td>religion</td>
<td>863</td>
<td>216</td>
<td>717</td>
<td>18822</td>
<td>18853</td>
</tr>
<tr>
<td>comp.</td>
<td>934</td>
<td>234</td>
<td>777</td>
<td>16282</td>
<td>10772</td>
</tr>
<tr>
<td>vote</td>
<td>1175</td>
<td>257</td>
<td>860</td>
<td>19813</td>
<td>43563</td>
</tr>
<tr>
<td>movie</td>
<td>1600</td>
<td>200</td>
<td>200</td>
<td>43800</td>
<td>49433</td>
</tr>
<tr>
<td>books</td>
<td>1440</td>
<td>360</td>
<td>200</td>
<td>21545</td>
<td>13806</td>
</tr>
<tr>
<td>dvd</td>
<td>1440</td>
<td>360</td>
<td>200</td>
<td>21086</td>
<td>13794</td>
</tr>
<tr>
<td>electr.</td>
<td>1440</td>
<td>360</td>
<td>200</td>
<td>10961</td>
<td>10227</td>
</tr>
<tr>
<td>kitch.</td>
<td>1440</td>
<td>360</td>
<td>200</td>
<td>9248</td>
<td>8998</td>
</tr>
</tbody>
</table>

Descriptive statistics of the datasets.
### Results

<table>
<thead>
<tr>
<th>dataset</th>
<th>no reg</th>
<th>lasso</th>
<th>ridge</th>
<th>elastic</th>
<th>OMP</th>
<th>group lasso</th>
<th>LDA</th>
<th>LSI</th>
<th>sen</th>
<th>GoW</th>
<th>w2v</th>
<th>GOMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>20NG</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>science</td>
<td>0.946</td>
<td>0.916</td>
<td>0.954</td>
<td>0.954</td>
<td>0.964*</td>
<td><strong>0.968</strong></td>
<td><strong>0.968</strong>*</td>
<td>0.942</td>
<td>0.967*</td>
<td><strong>0.968</strong></td>
<td>0.953*</td>
<td></td>
</tr>
<tr>
<td>sports</td>
<td>0.908</td>
<td>0.907</td>
<td>0.925</td>
<td>0.920</td>
<td>0.949*</td>
<td>0.959</td>
<td>0.964*</td>
<td><strong>0.966</strong></td>
<td>0.959*</td>
<td>0.946*</td>
<td>0.951*</td>
<td></td>
</tr>
<tr>
<td>religion</td>
<td>0.894</td>
<td>0.876</td>
<td>0.895</td>
<td>0.890</td>
<td>0.902*</td>
<td>0.918</td>
<td>0.907*</td>
<td><strong>0.934</strong></td>
<td>0.911*</td>
<td>0.916*</td>
<td>0.902*</td>
<td></td>
</tr>
<tr>
<td>computer</td>
<td>0.846</td>
<td>0.843</td>
<td>0.869</td>
<td>0.856</td>
<td>0.876*</td>
<td>0.891</td>
<td>0.885*</td>
<td>0.904</td>
<td>0.885*</td>
<td><strong>0.911</strong>*</td>
<td>0.902*</td>
<td></td>
</tr>
<tr>
<td>vote</td>
<td>0.606</td>
<td>0.643</td>
<td>0.616</td>
<td>0.622</td>
<td>0.684*</td>
<td>0.658</td>
<td>0.653</td>
<td>0.656</td>
<td>0.640</td>
<td>0.651</td>
<td><strong>0.687</strong>*</td>
<td></td>
</tr>
<tr>
<td>movie</td>
<td>0.865</td>
<td>0.860</td>
<td>0.870</td>
<td>0.875</td>
<td>0.860*</td>
<td><strong>0.900</strong></td>
<td>0.895</td>
<td>0.895</td>
<td>0.895</td>
<td>0.890</td>
<td>0.850</td>
<td></td>
</tr>
<tr>
<td>books</td>
<td>0.750</td>
<td>0.770</td>
<td>0.760</td>
<td>0.780</td>
<td>0.800</td>
<td>0.790</td>
<td>0.795</td>
<td>0.785</td>
<td>0.790</td>
<td>0.800</td>
<td><strong>0.805</strong>*</td>
<td></td>
</tr>
<tr>
<td>dvd</td>
<td>0.765</td>
<td>0.735</td>
<td>0.770</td>
<td>0.760</td>
<td>0.785</td>
<td>0.800</td>
<td>0.805*</td>
<td>0.785</td>
<td>0.795*</td>
<td>0.795*</td>
<td><strong>0.820</strong>*</td>
<td></td>
</tr>
<tr>
<td>electr.</td>
<td>0.790</td>
<td>0.800</td>
<td>0.800</td>
<td>0.825</td>
<td><strong>0.830</strong></td>
<td>0.800</td>
<td>0.815</td>
<td>0.805</td>
<td>0.820</td>
<td>0.815</td>
<td>0.800</td>
<td></td>
</tr>
<tr>
<td>kitch.</td>
<td>0.760</td>
<td>0.800</td>
<td>0.775</td>
<td>0.800</td>
<td>0.825</td>
<td>0.845</td>
<td><strong>0.860</strong>*</td>
<td>0.855</td>
<td>0.840</td>
<td>0.855*</td>
<td>0.830</td>
<td></td>
</tr>
</tbody>
</table>

Accuracy on the test sets. Bold font marks the best performance for a dataset, while * indicates statistical significance at $p < 0.05$ using micro sign test against lasso. For GOMP, we use w2v clusters and add all unigram features as individual groups.
## Sparsity

<table>
<thead>
<tr>
<th>dataset</th>
<th>no reg</th>
<th>lasso</th>
<th>ridge</th>
<th>elastic</th>
<th>OMP</th>
<th>LDA</th>
<th>LSI</th>
<th>sen</th>
<th>GoW</th>
<th>w2v</th>
<th>GOMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>20NG</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>science</td>
<td>100</td>
<td><strong>1</strong></td>
<td>100</td>
<td>63</td>
<td>2.7</td>
<td>19</td>
<td>20</td>
<td>86</td>
<td>19</td>
<td>21</td>
<td><strong>5.8</strong></td>
</tr>
<tr>
<td>sports</td>
<td>100</td>
<td><strong>1</strong></td>
<td>100</td>
<td>5</td>
<td>1.8</td>
<td>60</td>
<td>11</td>
<td>6.4</td>
<td>55</td>
<td>44</td>
<td><strong>7.7</strong></td>
</tr>
<tr>
<td>religion</td>
<td>100</td>
<td><strong>1.1</strong></td>
<td>100</td>
<td>3</td>
<td>1.5</td>
<td>94</td>
<td>31</td>
<td>99</td>
<td>10</td>
<td>85</td>
<td><strong>1.5</strong></td>
</tr>
<tr>
<td>computer</td>
<td>100</td>
<td>1.6</td>
<td>100</td>
<td>7</td>
<td><strong>0.6</strong></td>
<td>40</td>
<td>35</td>
<td>77</td>
<td>38</td>
<td>18</td>
<td><strong>4.9</strong></td>
</tr>
<tr>
<td>Sentiment</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>vote</td>
<td>100</td>
<td><strong>0.1</strong></td>
<td>100</td>
<td>8</td>
<td>5</td>
<td>15</td>
<td>16</td>
<td>13</td>
<td>97</td>
<td>13</td>
<td><strong>1.5</strong></td>
</tr>
<tr>
<td>movie</td>
<td>100</td>
<td>1.3</td>
<td>100</td>
<td>59</td>
<td><strong>0.9</strong></td>
<td>72</td>
<td>81</td>
<td>55</td>
<td>90</td>
<td>62</td>
<td><strong>2.3</strong></td>
</tr>
<tr>
<td>books</td>
<td>100</td>
<td><strong>3.3</strong></td>
<td>100</td>
<td>14</td>
<td>4.6</td>
<td>41</td>
<td>74</td>
<td>72</td>
<td>90</td>
<td>99</td>
<td><strong>8.3</strong></td>
</tr>
<tr>
<td>dvd</td>
<td>100</td>
<td>2</td>
<td>100</td>
<td>28</td>
<td>2.8</td>
<td>64</td>
<td>8</td>
<td>8</td>
<td>58</td>
<td>64</td>
<td><strong>9</strong></td>
</tr>
<tr>
<td>electr.</td>
<td>100</td>
<td>4</td>
<td>100</td>
<td>6</td>
<td>6.3</td>
<td>10</td>
<td>8</td>
<td>43</td>
<td>8</td>
<td>9</td>
<td><strong>12</strong></td>
</tr>
<tr>
<td>kitch.</td>
<td>100</td>
<td>4.5</td>
<td>100</td>
<td>79</td>
<td><strong>4.3</strong></td>
<td>73</td>
<td>44</td>
<td>27</td>
<td>75</td>
<td>46</td>
<td><strong>6.5</strong></td>
</tr>
</tbody>
</table>

Fraction (in %) of non-zero feature weights in each model for each dataset. Bold for best, blue for best group.
## Analysis

<table>
<thead>
<tr>
<th>dataset</th>
<th>GoW</th>
<th>word2vec</th>
</tr>
</thead>
<tbody>
<tr>
<td>science</td>
<td>79</td>
<td>691</td>
</tr>
<tr>
<td>sports</td>
<td>137</td>
<td>630</td>
</tr>
<tr>
<td>religion</td>
<td>35</td>
<td>639</td>
</tr>
<tr>
<td>computer</td>
<td>95</td>
<td>594</td>
</tr>
</tbody>
</table>

### Number of groups.

<table>
<thead>
<tr>
<th>dataset</th>
<th>lasso</th>
<th>ridge</th>
<th>elastic</th>
</tr>
</thead>
<tbody>
<tr>
<td>science</td>
<td>10</td>
<td>1.6</td>
<td>1.6</td>
</tr>
<tr>
<td>sports</td>
<td>12</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>religion</td>
<td>12</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>computer</td>
<td>7</td>
<td>1.4</td>
<td>0.8</td>
</tr>
</tbody>
</table>

### Features with the largest weights.

- **lasso**
  - space: orbit, space, contribute, funding, landing
  - hockey: nhl, hockey, playoffs, cup, wings

- **OMP**
  - space: space, orbit, moon, planets, scientifically
  - hockey: nhl, hockey, playoff, wings, cup

<table>
<thead>
<tr>
<th>dataset</th>
<th>lasso</th>
<th>ridge</th>
<th>elastic</th>
<th>group lasso</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LDA</td>
<td>LSI</td>
<td>sentence</td>
<td>GoW</td>
</tr>
<tr>
<td>science</td>
<td>15</td>
<td>11</td>
<td>76</td>
<td>12</td>
</tr>
<tr>
<td>sports</td>
<td>7</td>
<td>20</td>
<td>67</td>
<td>5</td>
</tr>
<tr>
<td>religion</td>
<td>10</td>
<td>4</td>
<td>248</td>
<td>6</td>
</tr>
<tr>
<td>computer</td>
<td>8</td>
<td>6</td>
<td>43</td>
<td>5</td>
</tr>
</tbody>
</table>

### Time (in seconds) for learning with best hyperparameters.

- **lasso**: 78 seconds
- **OMP**: 21 seconds
- **OMP**: 38 seconds
- **OMP**: 4 seconds
Selecting the number of atoms

Accuracy in dev set vs number of active atoms.
### Summary

#### Pros
- OMP requires no prior
- GOMP beats group lasso in some cases (but always in sparsity)
- Fast with relatively small number of dimensions

#### Cons
- Greedy algorithm → GOMP gets slow (adding single terms)
- Groups need to be good

#### Contributions
- Clusters of words can enhance structured regularization
- Introduce overlapping GOMP and compare it with group lasso
- Creating super-sparse models

#### Can we use these linguistic structures (groups) for more?
Yes for distances based on word embeddings!
Outline

1. Introduction
2. Context
3. Graph-based Representations
4. Structured Regularization
5. Sets & Distances
6. Conclusion
Distances in word embeddings

- Word embeddings to compute similarity between documents
- Cosine or euclidean distance
- Centroids lose a lot of info

Word Mover’s Distance.

- WMD and Supervised WMD (Kusner et al., 2015);(Huang et al., 2016)
- Optimal transport (Villani, 2008)
- SOTA distance-based classification
Enhancing tricks for WMD

Limitations

- Exact WMD scales at $O(n^3)$
  - Relaxed WMD
- Not all words are important
- Outlier words
- Labels not exploited

We focus on:

- Stopword removal
- Cross document-topic comparison by clustering word embeddings
- Convex metric learning
Contributions

Stopword removal
- More than 10 stopword lists
- SMART (Salton and Buckley, 1971)
- Gensim & spacy (Stone et al., 2010)

Convex metric learning
- Maximally Collapsing Metric Learning (Globerson and Roweis, 2006)
- Large Margin Nearest Neighbor (Weinberger and Saul, 2009)

Cross document-topic comparison

\[ \text{RWMD}(\text{Doc}2, \text{DT}1) \]
\[ \mu(d1,d2) \]
\[ \text{RWMD}(\text{Doc}1, \text{DT}2) \]

\[ \text{DT}1 = \text{Doc}1 \text{ words } + \text{Topics1} \]

The LMNN architecture.
Datasets & setup

Datasets

(1) **BBCSPORT**: sports articles between 2004-2005

(2) **Twitter**: sentiment tweets

(3) **Recipe**: recipes by region

(4) **Ohsumed**: cardiovascular medical abstracts

(5) **Classic**: academic papers by publisher

(6) **Reuters**: news topics

(7) **Amazon**: product reviews by sentiment

(8) **20News**: news articles in 20 categories

<table>
<thead>
<tr>
<th>Dataset</th>
<th>n</th>
<th>Voc</th>
<th>Unique Words(avg)</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>BBCSPORT</td>
<td>517</td>
<td>13243</td>
<td>117</td>
<td>5</td>
</tr>
<tr>
<td>Twitter</td>
<td>2176</td>
<td>6344</td>
<td>9.9</td>
<td>3</td>
</tr>
<tr>
<td>Recipe</td>
<td>3059</td>
<td>5708</td>
<td>48.5</td>
<td>15</td>
</tr>
<tr>
<td>Ohsumed</td>
<td>3999</td>
<td>31789</td>
<td>59.2</td>
<td>10</td>
</tr>
<tr>
<td>Classic</td>
<td>4965</td>
<td>24277</td>
<td>38.6</td>
<td>4</td>
</tr>
<tr>
<td>Reuters</td>
<td>5485</td>
<td>22425</td>
<td>37.1</td>
<td>8</td>
</tr>
<tr>
<td>Amazon</td>
<td>5600</td>
<td>42063</td>
<td>45.0</td>
<td>4</td>
</tr>
<tr>
<td>20NG</td>
<td>11293</td>
<td>29671</td>
<td>72</td>
<td>20</td>
</tr>
</tbody>
</table>

Datasets in our TC experiments.

Setup

- Pretrained w2v *(Mikolov et al., 2013)*
- Stopwords by *(Stone et al., 2010)* (used in Gensim & spacy)
- In k-means we set $k = 500$
## Results

<table>
<thead>
<tr>
<th></th>
<th>BBCsport</th>
<th>Twitter</th>
<th>Recipe</th>
<th>Ohsumed</th>
<th>Classic</th>
<th>Reuters</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSI</td>
<td>4.30 ± 0.60</td>
<td>31.70 ± 0.70</td>
<td>45.40 ± 0.50</td>
<td>44.20</td>
<td>6.70 ± 0.40</td>
<td>6.30</td>
</tr>
<tr>
<td>WMD</td>
<td>4.60 ± 0.70</td>
<td>28.70 ± 0.60</td>
<td>42.60 ± 0.30</td>
<td>44.50</td>
<td>2.88 ± 0.10</td>
<td>3.50</td>
</tr>
<tr>
<td>Stopword RWMD</td>
<td>4.27 ± 1.19</td>
<td>27.51 ± 1.00</td>
<td>43.98 ± 1.40</td>
<td>44.27</td>
<td>3.25 ± 0.50</td>
<td>5.25</td>
</tr>
<tr>
<td>All, 5nn</td>
<td>6.00 ± 1.34</td>
<td>29.23 ± 1.09</td>
<td>42.52 ± 1.18</td>
<td>46.73</td>
<td>3.18 ± 0.44</td>
<td>6.26</td>
</tr>
<tr>
<td>All, 5nn, Mean</td>
<td>4.00 ± 1.55</td>
<td>28.58 ± 2.29</td>
<td>42.53 ± 0.67</td>
<td>43.90</td>
<td>3.08 ± 0.62</td>
<td>5.76</td>
</tr>
<tr>
<td>k-means, 5nn</td>
<td>5.91 ± 2.65</td>
<td>28.56 ± 1.20</td>
<td>42.23 ± 1.15</td>
<td>46.50</td>
<td>2.98 ± 0.66</td>
<td>4.71</td>
</tr>
<tr>
<td>k-means, 5nn, Mean</td>
<td>3.82 ± 1.72</td>
<td>28.50 ± 1.51</td>
<td>41.95 ± 1.04</td>
<td>44.05</td>
<td>3.08 ± 0.51</td>
<td>4.57</td>
</tr>
<tr>
<td>S-WMD (NCA)</td>
<td>2.10 ± 0.50</td>
<td>27.50 ± 0.50</td>
<td>39.20 ± 0.30</td>
<td><strong>34.30</strong></td>
<td>3.20 ± 0.20</td>
<td>3.20</td>
</tr>
<tr>
<td>LMNN</td>
<td><strong>1.73 ± 0.67</strong></td>
<td>28.86 ± 2.22</td>
<td>40.88 ± 1.88</td>
<td>39.59</td>
<td><strong>2.76 ± 0.30</strong></td>
<td>4.02</td>
</tr>
<tr>
<td>MCML</td>
<td>2.45 ± 1.27</td>
<td><strong>27.15 ± 1.36</strong></td>
<td><strong>38.93 ± 1.24</strong></td>
<td>42.38</td>
<td>3.56 ± 0.49</td>
<td><strong>2.92</strong></td>
</tr>
</tbody>
</table>

Comparison in knn test error(%) to LSI, WMD and S-WMD. Blue shows best results in unsupervised methods and bold indicates best result for a dataset.
Summary

Contributions

- many stopword lists discovered
- adding neighbour words via clustering helps
- convex metric learning can boost accuracy

Can we do more?

Learning document representations via distances in word embeddings.

Bipartite Graph Matching for two documents.
### Sets in word embeddings

#### What is a set?
Complex data sets composed of simpler objects.

→ NLP: documents as sets of word embeddings

#### Standard machine learning algorithms:
- fixed dimensional data instances
- data representations and learning are independent

#### Text classification as set classification
1. distance/similarity measure or kernel that finds a correspondence between each pair of sets
2. use instance-based machine learning algorithm (knn or SVM)

← high computational and memory complexity (all to all comparison)
Main approaches

**Neural networks:**
- PointNet (Qi et al., 2017) and DeepSets (Zaheer et al., 2017) transform the vectors of the sets into new representations, then apply permutation-invariant functions
- unordered sets $\rightarrow$ ordered sequences $\rightarrow$ RNN (Vinyals et al., 2015)

**Kernels:**
- estimate a probability distribution on each set of vectors, then derive their similarity using distribution-based comparison measures such as Fisher kernels (Jaakkola and Haussler, 1999)
- map the vectors to multi-resolution histograms, then compare them with a weighted histogram intersection measure to find an approximate correspondence (Grauman and Darrell, 2007)

**Distance metric learning:**
- learning a distance function over objects
- text $\rightarrow$ Supervised Word Mover’s Distance (Huang et al., 2016)
## Pros & cons

### Advantages
- easily extended (NNs)
- fast and scalable with GPUs (NNs)
- very effective in several tasks (NNs & kernels)

### Limitations
- simple permutation invariant functions
- limits the expressive power of these architectures
- kernels ⇒ high computational complexity
- data representation and learning are independent from each other
Contributions

- **RepSet** a novel architecture for performing machine learning on sets which, in contrast to traditional approaches, is capable of adapting data representation to the task at hand

- **ApproxRepSet**, a simplified architecture which can be interpreted as an approximation of the proposed model, able to handle very large datasets

- Evaluation of the proposed architecture on several benchmark datasets in text classification
Learning sets via distances in word embeddings

RepSet: Neural Networks for Learning Set Representations

Each element of the input set is compared with the elements of all “hidden sets”, and the emerging matrices serve as the input to bipartite matching. The values of the BM problems correspond to the representation of the input set.
Given an input set of vectors, \( X = \{ \mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{k_1} \} \) and a hidden set \( Y_i = \{ \mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_{k_2} \} \), first we identify which of the two sets has the highest cardinality. If \( |X| \geq |Y_i| \), we solve the following linear program:

\[
\max \sum_{i=1}^{k_1} \sum_{j=1}^{k_2} x_{ij} f(\mathbf{v}_i, \mathbf{u}_j) \text{ subject to:} \\
\sum_{i=1}^{k_1} x_{ij} \leq 1 \quad \forall j \in \{1, \ldots, k_2\} \\
x_{ij} \geq 0 \quad \forall i \in \{1, \ldots, k_1\}, \forall j \in \{1, \ldots, k_2\}
\] (13)

- multiple elements of \( Y_i \) (the bigger set) can be matched with the same element of \( X \)
- the optimal solution matches an element of \( Y \) with one of \( X \) if their inner product is positive and is the highest among the inner products between all the pairs
## Results

<table>
<thead>
<tr>
<th></th>
<th>BBCSPORT</th>
<th>TWITTER</th>
<th>RECIPE</th>
<th>OHSUMED</th>
<th>CLASSIC</th>
<th>REUTERS</th>
<th>AMAZON</th>
<th>20NG</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSI</td>
<td>4.30 ± 0.60</td>
<td>31.70 ± 0.70</td>
<td>45.40 ± 0.50</td>
<td>44.20</td>
<td>6.70 ± 0.40</td>
<td>6.30</td>
<td>9.30 ± 0.40</td>
<td>28.90</td>
</tr>
<tr>
<td>WMD</td>
<td>4.60 ± 0.70</td>
<td>28.70 ± 0.60</td>
<td>42.60 ± 0.30</td>
<td>44.50</td>
<td><strong>2.88</strong> ± 0.10</td>
<td>3.50</td>
<td>7.40 ± 0.30</td>
<td>26.80</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>S-WMD</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2.10 ± 0.50</td>
<td>27.50 ± 0.50</td>
<td>39.20 ± 0.30</td>
<td>34.30</td>
<td>3.20 ± 0.20</td>
<td>3.20</td>
<td>5.80 ± 0.10</td>
<td>26.80</td>
</tr>
<tr>
<td>DeepSets</td>
<td>25.45 ± 20.1</td>
<td>29.66 ± 1.62</td>
<td>70.25</td>
<td>71.53</td>
<td>5.95 ± 1.50</td>
<td>10.00</td>
<td>8.58 ± 0.67</td>
<td>38.88</td>
</tr>
<tr>
<td>NN-mean</td>
<td>10.09 ± 2.62</td>
<td>31.56 ± 1.53</td>
<td>64.30 ± 7.30</td>
<td>45.37</td>
<td>5.35 ± 0.75</td>
<td>11.37</td>
<td>13.66 ± 3.16</td>
<td>38.40</td>
</tr>
<tr>
<td>NN-max</td>
<td>2.18 ± 1.75</td>
<td>30.27 ± 1.26</td>
<td>43.47 ± 1.05</td>
<td>35.88</td>
<td>4.21 ± 0.11</td>
<td>4.33</td>
<td>7.55 ± 0.63</td>
<td>32.15</td>
</tr>
<tr>
<td>NN-attention</td>
<td>4.72 ± 0.97</td>
<td>29.09 ± 0.62</td>
<td>43.18 ± 1.22</td>
<td><strong>31.36</strong></td>
<td>4.42 ± 0.73</td>
<td>3.97</td>
<td>6.92 ± 0.51</td>
<td>28.73</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>RepSet</th>
<th>Approx</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2.00 ± 0.89</td>
<td><strong>25.42</strong> ± 1.10</td>
<td><strong>38.57</strong> ± 0.83</td>
<td><strong>33.88</strong></td>
<td>3.38 ± 0.50</td>
<td>3.15</td>
<td><strong>5.29</strong> ± 0.28</td>
<td><strong>22.98</strong></td>
</tr>
</tbody>
</table>

Classification test error of the proposed architecture and the baselines.

<table>
<thead>
<tr>
<th>BBCSPORT</th>
<th>Points of sets</th>
<th>Centroids</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>cup, club, united, striker, arsenal</td>
<td>Modric</td>
</tr>
<tr>
<td>2</td>
<td>scrum, nations, scotland, ireland, france</td>
<td>rugby</td>
</tr>
<tr>
<td>3</td>
<td>winner, olympic, tennis, court, Olympic_gold_medalist</td>
<td>Olympic_Medalist</td>
</tr>
<tr>
<td>4</td>
<td>captain, player, striker, ball, game</td>
<td>skipper</td>
</tr>
<tr>
<td>5</td>
<td>wickets, series, cricket, bat, side</td>
<td>batsmen</td>
</tr>
</tbody>
</table>

Terms of the employed pre-trained model that are most similar to the points and centroids of the elements of 5 hidden sets.
Summary

Contributions:

- RepSet, neural networks for learning set representations
  - exhibits powerful permutation invariance properties
  - highly interpretable
  - easily extended to deeper architectures
- introduced a relaxed version (ApproxRepSet)
  - involves fast matrix operations and scales to large datasets
- effective on text categorization
- PyTorch implementation (fast in GPU)

Future work:

- replacing matching with optimal transport?
- other tasks?
  - graph mining: graphs as sets of node embeddings
  - computer vision: images as sets of local features
Outline

1. Introduction
2. Context
3. Graph-based Representations
4. Structured Regularization
5. Sets & Distances
6. Conclusion
In this thesis, I have:

- Explored in details the fields of NLP and ML
- Worked in methods to exploit context and discover latent concepts
- Tested in real world tasks
Ph.D. thesis contributions

▶ Created new **graph-based representations** to model text in more meaningful structures and derived innovative **metrics** out of them

▶ Extracted novel **linguistic groups** and developed new methods for **structured regularization**

▶ Boosted existing **document comparison** techniques and designed a new **set representation learning** model via distances

▶ Applied them to text classification
Future work

- **TW-IDW**: Graph-of-Documents instead of a Bag-of-Documents in order for instance to compute an alternative to IDF

- **Graph neural networks**: involves neural message passing

- **Linguistic structured attention**: a group lasso attention mechanism for deep learning architectures

- **Adversarial structured regularization**

- **Gradient-based learning for WMD**

Thank you!
References


Deshpande, Mukund et al. (2005). “Frequent Substructure-Based Approaches for Classifying Chemical Compounds”. In: *IEEE Trans. on Knowl. and Data Eng.* TKDE 17.8, pp. 1036–1050.


Outsios, Stamatis et al. (2018). “Word Embeddings from Large-Scale Greek Web content”. In: Spoken Language Technology (SLT).


Salton, Gerard and C Buckley (1971). The SMART information retrieval system.

Shanavas, Niloofer et al. (2016). “Centrality-Based Approach for Supervised Term Weighting”. In: Data Mining Workshops (ICDMW), 2016 IEEE 16th International Conference on. ICDM. IEEE, pp. 1261–1268.


Wang, Rui, Wei Liu, and Chris McDonald (2015). “Corpus-independent Generic Keyphrase Extraction Using Word Embedding Vectors”. In:

References VI

Yao, Liang, Chengsheng Mao, and Yuan Luo (2019). “Graph Convolutional Networks for Text Classification”. In: Association for the Advancement of Artificial Intelligence.


### Regularizer examples

**Examples with LSI regularizer.**

<table>
<thead>
<tr>
<th>= 0</th>
<th>islands inta spain galapagos canary originated anodise advertises jewelry mercedes benzes diamond trendy octave chanute lillienthal</th>
</tr>
</thead>
<tbody>
<tr>
<td>≠ 0</td>
<td>vibrational broiled relieving succumb spacewalks dna nf-psychiatry itself commented usenet golded insects alternate self-consistent retrospect</td>
</tr>
</tbody>
</table>

**Examples with word2vec regularizer.**

<table>
<thead>
<tr>
<th>= 0</th>
<th>village town edc fashionable trendy trendy fashionable points guard guarding crown title champion champions</th>
</tr>
</thead>
<tbody>
<tr>
<td>≠ 0</td>
<td>numbness tingling dizziness fevers laryngitis bronchitis undergo undergoing undergoes undergone healed mankind humanity civilization planet nasa kunin lang tao kay kong</td>
</tr>
</tbody>
</table>

**Examples with graph-of-words regularizer.**

<table>
<thead>
<tr>
<th>= 0</th>
<th>piscataway combination <a href="mailto:jil@donuts0.uucp">jil@donuts0.uucp</a> jamie reading/seeing chambliss left-handedness abilities lubin acad sci obesity page erythromycin bottom</th>
</tr>
</thead>
<tbody>
<tr>
<td>≠ 0</td>
<td>and space the launch health for use that medical you space cancer and nasa hiv health shuttle for tobacco that cancer that research center space hiv aids are use theory keyboard data telescope available are from system information space ftp</td>
</tr>
</tbody>
</table>
Algorithm 1 Logistic Overlapping GOMP

**Input:** $X = [x_1, \ldots, x_N]^T \in \mathbb{R}^{N \times d}$, $y \in \{-1, 1\}^N$, $\{G_1, \ldots, G_J\}$ (group structure), $K$ (budget), $\epsilon$ (precision), $\lambda$ (regularization factor).

**Output:** $I = \emptyset$, $r^{(0)} = y$, $k = 1$

1: while $|I| \leq K$ do
2: 
3: 
4: 
5: end if
6: $I = I \cup \{G_{j(k)}\}$
7: for $i = 1$ to $J$ do
8: 
9: end for
10: $\theta^{(k)} = \arg\min_{\theta} \sum_{i=1}^{N} \mathcal{L}(x_i, \theta, y_i) + \lambda \|\theta\|_2^2$ s.t. $\text{supp}(\theta) \subseteq I$
11: $r^{(k)} = \frac{1}{1 + \exp\{-X^T \theta^{(k)}\}} - 1 \{y\}$
12: $k += 1$
13: end while
14: return $\theta^{(k)}, I$
Overlapping GOMP
Difference with GOMP

- Overlapping GOMP extends the standard GOMP in the case where the groups of indices are overlapping, i.e. $G_i \cap G_j \neq \emptyset$ for $i \neq j$.

- The main difference with GOMP is that each time a group becomes active, we remove its indices from each inactive group: $G_i = G_i \setminus G_{j(k)}$, $\forall i \in \{1, \ldots, J\}$.

- In this way, the theoretical properties of GOMP hold also in the case of the overlapping GOMP.
Computing the gradients (1/2)

The gradient of the loss function with respect to the \( j^{th} \) row of the weight matrix of the penultimate layer is:

\[
\frac{\partial L}{\partial W_j^{(c)}} = p_j - y_j \tag{14}
\]

We next define the following differentiable function:

\[
g(D^{(k)*}, W^{(k)}) = \sum_{i=1}^{k_1} \sum_{j=1}^{k_2} D_{ij}^{(k)*} f(v_i, u_j^{(k)}) \tag{15}
\]

where \( D^{(k)*} \) is the matrix that contains the optimal values of the variables of the BM problem for the corresponding hidden set \( Y_k \).

\[
\frac{\partial}{\partial W^{(k)}} g(D^{(k)*}, W^{(k)}) = \frac{\partial}{\partial W^{(k)}} tr(D^{(k)*\top} G^{(k)}) \tag{16}
\]
Computing the gradients (2/2)

We have:

- \( X \), a matrix whose rows correspond to the elements of set \( X \).
- \( G^{(k)} = \text{ReLU}(XW) \), weights that have been set to zero during the forward pass are stored as zero values in the optimal solution \( D^{(k)*} \).
- Indeed, no edge was created whenever the value of the dot product was negative. Then, none of these pairs have been used during the forward pass.

This yields:

\[
\frac{\partial}{\partial W^{(k)}} g(D^{(k)*}, W^{(k)}) = \frac{\partial}{\partial W^{(k)}} \text{tr}(D^{(k)*\top}XW^{(k)}) = X^\top D^{(k)*}
\]

Which finally gives:

\[
\frac{\partial L}{\partial W^{(k)}} = (p_k - y_k) \cdot X^\top D^{(k)*}
\]