Automated Bias Detection in Journalism

Richard Lee, James Chen, Jason Cho

**MOTIVATION**
- Be able to detect bias from parallel corpora (collections of machine-readable texts)
- To the best of our knowledge, automated bias detection has never been done.
- Previous research in NLP, e.g., annotation systems and sarcasm detection. Applications in journalism, politics, rhetoric, linguistics, law, education.
  - Increased objectivity in news articles and educational resources
  - Assisted grading and bias-flagging

**BACKGROUND**
- Some existing research directed towards related areas:
  - Lexical bias indicators \(^1\)
  - Quantification of media bias using conservative and liberal blogs \(^2\)
  - Automated annotation \(^3\)
  - Detection of other linguistic constructs, such as sarcasm \(^4\)
- However, no existing research directly tackling automated bias detection.
- Borrowed data (with permission) from "Shedding light on... biased language"
- Variety of NLP suites already popular, e.g., NLTK
- Consulted with Professor Eric K. Meyer, of the UIUC College of Media.
  - Topic framing, e.g., estate tax vs. death tax.
  - Choice of source, such as cherry-picked quotes and data points.
  - Choice of words, e.g., fetus vs. unborn baby.

**PROBLEM STATEMENT**
- Assuming we're given parallel corpora, identify biased sentences. From there, identify biased articles.
- We assume that we've already given annotated corpora to play with. In the future and in practice, we won't have such a luxury.
- Data borrowed from earlier study \(^5\):
  - Parallel topics: 2008 political atmosphere
  - Drawn from political blog posts.
- Match sentences/articles with three types of bias: liberal, neutral, or conservative.

**APPROACH**
- Used existing software to streamline design:
  - AlchemyAPI (for simple sentiment analysis and named entity recognition)
  - SVM \(^6\) handles classification of texts:
    - In lay terms, it does the guess work, provided it is given sufficient training data (patterns to build off).
- Bag of words approach: all the unique words in a document are given an index. Documents then reconstructed from indexes as vectors, e.g:
  - John likes to watch movies. Mary likes too. John also likes to watch football games.
  - {"John": 1, "likes": 2, "to": 3, "watch": 4, "movies": 5, "also": 6, "football": 7, "games": 8, "Mary": 9, "too": 10}[
  - [1, 2, 1, 1, 0, 0, 8, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0] \(^5\)

**RESULTS, CONT.**

<table>
<thead>
<tr>
<th>Bias Type</th>
<th>Positive</th>
<th>Neutral</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percentage</td>
<td>0%</td>
<td>100%</td>
<td>0%</td>
</tr>
</tbody>
</table>

**RESULTS**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Error</td>
<td>39%</td>
<td>39.9%</td>
<td>39%</td>
<td>61.3%</td>
</tr>
</tbody>
</table>

Table. Errors from classic bag-of-words experiments during textual classification vs. how machines overlapped classification samples present in all datasets

**CONCLUSION**
- Three categories, baseline accuracy is 33.33% (assuming completely random).
- Using simple bag-of-words approach achieved ~40% accuracy for three-way classification.
- Using bag of words approach for binary classification, we achieve ~60% accuracy.

**FUTURE WORKS**
- Bias != subjectivity
  - E.g., "The second amendment protects gun owners’ rights in America," versus "I think Inception was quite overrated."
- More advanced bias detection techniques
  - Smarter "dictionary" of very biased topics
    - I.e., creating a list of words that give their sentences added priority when weighing sentiment
  - Summing sentiment by topic, etc.

**REFERENCES**