Text Representation: The Foundation of Any Text Mining Task
Text data...

... is the most natural way of encoding human knowledge

☐ Scientific literature
☐ Manuals
☐ News

... is by far the most common type of information encountered by people

☐ What data have you produced and consumed today?

... is the most expressive form of information

☐ It can even describe other media such as video or images
☐ How am I communicating right now?
Text: Input to many applications

First step: Feature extraction ($n$-gram words)

A dog is chasing a boy on the playground.
First step: Feature extraction (POS tags)

\[ A_{DT} \text{ dog}_{NN} \text{ is } V_{VBZ} \text{ chasing}_{VBG} a_{DT} \text{ boy}_{NN} \text{ on}_{IN} \text{ the}_{DT} \text{ playground}_{NN} . \]
First step: Feature extraction (parse trees)

S

NP

DT A NN dog

VP

VBZ is

PPPP

VP

VBG chasing

NP

chasing

PP

IN on

NP

da boy

NP

the playground
First step: Feature extraction (parse trees)

```
S
├── NP
│   ├── NNP
│       └── Cuba
├── VP
│   ├── VBZ
│       └── is
├── VP
│   ├── VBG
│       └── chasing
│   └── NP
│       ├── PP
│           ├── DT
│               └── the
│           └── NP
│               ├── NN
│                   └── the
│               └── IN
│                   └── on
│       └── NP
│           ├── DT
│               └── a
│           └── NN
│               └── boy
```
First step: Feature extraction (entities/relations)

A dog $\rightarrow_{chase}$ a boy $\rightarrow_{on}$ the playground

\[ \text{Animal} \rightarrow_{\text{chase}} \text{Person} \rightarrow_{\text{on}} \text{Location} \]
First step: Feature extraction (entities/relations)

Cuba \(\rightarrow\text{chase}\) a boy \(\rightarrow\text{on}\) the playground
Using features

Feature vector similarity
- Information retrieval
- Clustering

Feature presence, absence, and co-occurrence
- Topic modeling
- Pattern/association mining

Sequences of features
- Information extraction
- NLP tools
Anne Hathaway is fighting an impossible battle against her haters. It’s not worth her time.

Oh, the case of poor Anne Hathaway. The Oscar-winning actress has received an avalanche of bad press over the last couple of years for that most polarizing kind of offense: Being herself.
Feature vector

\{battle, hathaway, impossible, time, worth, \ldots\}
Feature efficacy

Does Anne Hathaway News Drive Berkshire Hathaway's Stock?

Given the awesome correlating powers of today's stock trading computers, the idea may not be as far-fetched as you think.

We want features that are powerful, efficient, interpretable, and general.
“I saw this movie last night after being coaxed to by a few friends of mine. I’ll admit that I was reluctant to see it because from what I knew of Ashton Kutcher he was only able to do comedy. I was wrong.”

Was the opinion of this movie positive or negative?
"I agree this opinion that part-time job is important for college school students, because I studied a lot of thing with my part-time job. At first, the communication skill is necessary to work."

Was this essay written by a Chinese, Japanese, or English student?
“The smoking people should think a lot how smoking can cause the problem to other person beside you.”

How can we make this sentence sound more fluent?
Batch essay grading

Given thousands of essays, how can we automatically analyze them for mastery of English?
Two feature representations

**Structural parse tree features**: characterize the structure of grammatical production rules, ignoring rule labels.

**SyntacticDiff**: compare current text to a reference corpus and characterize the differences.

Power, efficiency, interpretability, and generality are major concerns.
Motivation: Structural parse tree features

Simple lexical features + deep syntactic features = best performance\(^3\).

Can we generalize “deep syntactic features” even more?

Designing a more general feature would help with data sparseness and overfitting.

Grammatical parse tree

```
S
  ├── NP
  │    ├── PRP
  │    │    └── They
  │    └── VBP
  │         └── have
  └── VP
      └── NP
          └── JJ
              └── JJ
                  └── NNS
                      └── many
                          └── theoretical
                              └── ideas
```
"They have many theoretical ideas."

---

Structural parse tree features: Semi-skeletons

“They have many theoretical ideas.”
Structural parse tree features: Skeletons

“They have many theoretical ideas.”
Classification results: Structural parse tree features

Together with unigram words, structural features improved:
- Sentiment analysis, taking accuracy from 82% to 82.8%
- Nationality detection, taking accuracy from 91.6% to 94.2%

The increased performance shows that the new features capture an orthogonal perspective of text compared to existing features.

Feature selection showed that the most informative features were a mix of words and tree structures.
Discussion: Structural parse tree features

Power
- Improved classification performance

Efficiency
- Less sparse → fewer features → less computation
- Parsing algorithm itself can be simplified to not include syntactic category types

Interpretability
- Complexity of structure visualized without label distractors
- Common substructure occurrences

Generality
- Derive features from any tree structure: XML, DOM, ...
- Applicable to other areas aside from categorization
Motivation: SyntacticDiff

“The smoking people should think a lot how smoking can cause the problem to other person beside you.”

```
remove(the, 12)
substitute(problem → problems, 13)
substitute(to → for, 14)
substitute(person → people, 16)
```

“The smoking people should think a lot how smoking can cause problems for other people beside you.”
Most existing work on text object representation assumes features are derived solely from the object itself.

Comparing a text object to an out-of-corpus reference reveals differences that might not be discernable otherwise.

Differences from the reference can transform (or, “correct”) the text object, making it appear more similar to the reference.

Or, the differences themselves can represent the object.

SyntacticDiff: Where to edit?

\[ \text{insert}(S, w, j) \quad \text{remove}(S, j) \quad \text{substitute}(S, w, j) \]

\[ j = \arg\max_{i \in [0,m]} \{ \text{Perp}(w_i, w_{i+1}, \ldots, w_{i+n-1}) \} \]

\[ \text{Perp(} \text{the problem to}) = 53, i = 11 \]
\[ \text{Perp(} \text{problem to other}) = 67, i = 12 \]
\[ \text{Perp(} \text{to other person}) = 144, i = 13 \]

SyntacticDiff: Weighted edits

\[ S^* = \arg \min_{S \in \text{candidates}} \{ \alpha \cdot \text{Perp}(S) + \beta \cdot W_S \} \]

\[ W_S = \frac{1}{|S_E|} \sum_{\text{edit} \in S_E} \text{penalty}(\text{edit}) \]

\[ S_E = \{ \text{insert(\text{the})}, \text{remove(\text{car})}, \ldots, \text{sub(\text{a} \rightarrow \text{an})} \} \]
Summarization task

For a MOOC or non-native text mining application:

- find groups of students that make similar errors
- target problem areas depending on the group of students
- pair students with complementary strengths and weaknesses
- batch grading
- ...

Results: Summarization with bigram words

<table>
<thead>
<tr>
<th>topic 1</th>
<th>topic 4</th>
<th>topic 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>smoke cigarette</td>
<td>restaurant owners</td>
<td>if you</td>
</tr>
<tr>
<td>smoking zone</td>
<td>smoking bans</td>
<td>you are</td>
</tr>
<tr>
<td>can make</td>
<td>bars and</td>
<td>you can</td>
</tr>
<tr>
<td>sick in</td>
<td>customers would</td>
<td>when you</td>
</tr>
<tr>
<td>global warming</td>
<td>or non</td>
<td>you smoke</td>
</tr>
<tr>
<td>this policy</td>
<td>ban on</td>
<td>for your</td>
</tr>
<tr>
<td>public space</td>
<td>in bars</td>
<td>around you</td>
</tr>
<tr>
<td>make many</td>
<td>smoke filled</td>
<td>yourself and</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>topic 12</th>
<th>topic 15</th>
</tr>
</thead>
<tbody>
<tr>
<td>the media</td>
<td>passive smoker</td>
</tr>
<tr>
<td>harmful for</td>
<td>active smoker</td>
</tr>
<tr>
<td>responsibility of</td>
<td>active smokers</td>
</tr>
<tr>
<td>hotels or</td>
<td>in indonesia</td>
</tr>
<tr>
<td>cigarette companies</td>
<td>the active</td>
</tr>
<tr>
<td>have shown</td>
<td>can disturb</td>
</tr>
<tr>
<td>and teenagers</td>
<td>more dangerous</td>
</tr>
<tr>
<td>or anything</td>
<td>all restaurant</td>
</tr>
</tbody>
</table>
Results: Summarization with SyntacticDiff

<table>
<thead>
<tr>
<th>topic 1</th>
<th>topic 4</th>
<th>topic 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>insert(the)</td>
<td>insert(a)</td>
<td>remove(you)</td>
</tr>
<tr>
<td>remove(opinion)</td>
<td>remove(so)</td>
<td>insert(to)</td>
</tr>
<tr>
<td>remove(cigarettes)</td>
<td>sub(lung→lungs)</td>
<td>sub(reason→reasons)</td>
</tr>
<tr>
<td>sub(give→giving)</td>
<td>sub(make→makes)</td>
<td>sub(ban→banning)</td>
</tr>
<tr>
<td>remove(bans)</td>
<td>remove(healthy)</td>
<td>remove(us)</td>
</tr>
<tr>
<td>insert(you)</td>
<td>remove(reasons)</td>
<td>remove(person)</td>
</tr>
<tr>
<td>remove(totally)</td>
<td>remove(as)</td>
<td>insert(are)</td>
</tr>
<tr>
<td>sub(cause→causes)</td>
<td>remove(even)</td>
<td>remove(better)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>topic 12</th>
<th>topic 15</th>
</tr>
</thead>
<tbody>
<tr>
<td>remove(so)</td>
<td>remove(area)</td>
</tr>
<tr>
<td>insert(for)</td>
<td>sub(seat→seats)</td>
</tr>
<tr>
<td>insert(in)</td>
<td>remove(of)</td>
</tr>
<tr>
<td>remove(not)</td>
<td>sub(stop→stopped)</td>
</tr>
<tr>
<td>sub(have→having)</td>
<td>remove(again)</td>
</tr>
<tr>
<td>remove(nonsmoker)</td>
<td>insert(i)</td>
</tr>
<tr>
<td>remove(that)</td>
<td>remove(all)</td>
</tr>
<tr>
<td>remove(increasing)</td>
<td>insert(it)</td>
</tr>
</tbody>
</table>

Documents are represented as their edits.
Discussion: SyntacticDiff

Power
- Native language identification: took accuracy from 81.8% to 85.9%.
- Grammatical error correction: places in top third in shared task.
- Summarization: better clusters than $n$-gram words.

Efficiency
- No deep NLP required.

Interpretability
- Each edit feature carries explicit meaning.

Generality
- Edit operations, penalties, and weights are all customizable.
Applications

 Structural tree features for
  □ Examining ASTs for code plagiarism/duplication (+ symbols)
  □ DOM structure for Web page clustering (+ text)

Using SyntacticDiff because I want...
  □ ... to write like Edgar Allan Poe.
  □ ... my cover letter to be professional.
  □ ... my research paper to sound scholarly.
  □ ... $X$ to be more like $Y$. 
Features that capture style

Quantify style as edit differences or tree structure usage—things that unigram words can’t capture.

A sudden style change in a stream of text could signal an event.

- **Text**: Campaign speeches
  - **Event**: The candidate drops out

- **Text**: Mark Zuckerberg’s Facebook posts
  - **Event**: Acquisition (Instagram, WhatsApp, Oculus…)

- **Text**: Blog from company’s leadership
  - **Event**: Bad earnings report

The best part: we don’t need explicit training data!
Text representation

Text data is the most natural way of encoding human knowledge.

Text mining tasks heavily depend on text feature extraction.

We discussed two novel methods for text representation:

- Structural parse tree features
- SyntacticDiff

Thanks!