Computational approaches to pun detection and interpretation

Tristan Miller

16th International Summer School and Symposium on Humour and Laughter
Transilvania University of Brașov
7 July 2016
Introduction

- Pun: a form of (humorous) wordplay in which a term suggests two meanings by exploiting a similarity in form

Where do otters keep their money? At the bank!
Scholarly study of puns

- Long history in rhetorical and literary criticism
- Now respectable in linguistics and cognitive sciences
Scholarly study of puns

- Long history in rhetorical and literary criticism
- Now respectable in linguistics and cognitive sciences
- Provides evidence for the psychological reality of linguistic phenomena
- Provides evidence for speakers’ awareness of linguistic processes
Scholarly study of puns

- Long history in rhetorical and literary criticism
- Now respectable in linguistics and cognitive sciences
- Provides evidence for the psychological reality of linguistic phenomena
- Provides evidence for speakers’ awareness of linguistic processes
- Computational humour and puns
  - Pun generation
  - Phonological analysis of puns
Scholarly study of puns

- Long history in rhetorical and literary criticism
- Now respectable in linguistics and cognitive sciences
- Provides evidence for the psychological reality of linguistic phenomena
- Provides evidence for speakers’ awareness of linguistic processes
- Computational humour and puns
  - Pun generation
  - Phonological analysis of puns
  - Detection and interpretation of puns
Overview of this talk

1. Motivation

2. Tasks in computational pun processing
   2.1 Pun detection
   2.2 Pun location
   2.3 Pun interpretation (including recovery of the target form)

3. Conclusions and future directions
Motivation: Human–computer interaction (HCI)

- “Humanization” of natural language interfaces
- Humorous interfaces increase user satisfaction without adversely affecting user efficiency
- Interfaces implementing wordplay and punning benefit augmentative and alternative communication
- Natural language understanding needed to move beyond canned and generated humour
Motivation: Sentiment analysis

- Sentiment analysis: automatically identify subjective information in text
- Useful in social research to track popular opinions and attitudes, and those of influencers
- Puns are particularly common in advertising
Motivation: Digital humanities

- Wordplay is a perennial topic in literary criticism and analysis
- Shakespeare’s puns among the most intensively studied aspects of his rhetoric
- Puns in historical literature often non-obvious due to diachronic shifts in semantics and pronunciation, obscure cultural references, etc.
- Digital humanities: computer-assisted analysis of literature
Motivation: Machine-assisted translation

- Comedic movies and TV shows among today’s most widely translated popular discourses
- Puns a recurrent, expected feature
- Challenges to translators:
  - Recognition of pun
  - Comprehension of pun
  - Selection and implementation of translation strategy
- MT systems could flag puns and propose ambiguity-preserving alternatives
Puns: Definition and classification

- Puns are a form of wordplay where a signifier suggests two meanings by exploiting a formal similarity.
Puns: Definition and classification

- Puns are a form of wordplay where a signifier suggests two meanings by exploiting a formal similarity
- Signifier can be any meaning-bearing phonological or orthographic sequence
Puns are a form of wordplay where a signifier suggests two meanings by exploiting a formal similarity.

Signifier can be any meaning-bearing phonological or orthographic sequence.

Relationship between the surface pun and the latent target:

<table>
<thead>
<tr>
<th>homophonic</th>
<th>heterophonic</th>
</tr>
</thead>
<tbody>
<tr>
<td>homographic</td>
<td>A political prisoner is one who stands behind her <em>convictions</em>.</td>
</tr>
<tr>
<td></td>
<td>A lumberjack’s world revolves on its <em>axes</em>.</td>
</tr>
<tr>
<td>heterographic</td>
<td>She fell through the window but felt no <em>pane</em>.</td>
</tr>
<tr>
<td></td>
<td>The sign at the nudist camp read, “<em>Clothed</em> until April.”</td>
</tr>
</tbody>
</table>
Puns: Definition and classification

- **Homographic**: same spelling
- **Heterographic**: different spelling
- **Homophonic**: same pronunciation
- **Heterophonic**: different pronunciation
Puns: Definition and classification

- **Homographic**: same spelling
- **Heterographic**: different spelling
- **Homophonic**: same pronunciation
- **Heterophonic**: different pronunciation
- **Homonymic, perfect**: synonyms for “homophonic” or “homographic” (or sometimes “homophonic and homographic”)
- **Heteronymic, paronymic, paronomastic, imperfect**: synonyms for “non-homonymic”
Computational processing of puns

- **Pun detection**: Given some text, does it contain a pun?
Computational processing of puns

- **Pun detection**: Given some text, does it contain a pun?
- **Pun location**: Given some text known to contain a pun, which part is the pun?
Computational processing of puns

- **Pun detection**: Given some text, does it contain a pun?
- **Pun location**: Given some text known to contain a pun, which part is the pun?
- **Pun interpretation**: Given some text known to contain a pun, and the location of the pun, what are the meanings of the pun and its target?
Pun detection

- Task: Given some text, does it contain a pun?
- A special case of humour detection
Pun detection

- Task: Given some text, does it contain a pun?
- A special case of humour detection
- General semantic incongruity detection (Mihalcea & Strapparava, 2005, 2006; Mihalcea & Pulman, 2007)
- Detecting a specific class of ambiguity-exploiting joke (Kiddon & Brun, 2011)
- Both of the above approaches rely on machine learning
Machine learning for joke detection

---

**Training data**

- Jokes
- Non-jokes

**Feature extraction**

**Learning algorithm**

---

**Test data**

- Jokes and non-jokes

**Feature extraction**

**Classifier**

- Jokes
- Non-jokes

---

7 July 2016 | Ubiquitous Knowledge Processing Lab | Department of Computer Science | Tristan Miller | 13
Machine learning for joke detection

Training data

Jokes
Non-jokes

Feature extraction

Learning algorithm

Test data

Jokes and non-jokes

Feature extraction

Classifier

Jokes
Non-jokes
Machine learning for joke detection

**Training data**
- Jokes
- Non-jokes

**Feature extraction**

**Learning algorithm**

**Test data**
- Jokes and non-jokes

**Feature extraction**

**Classifier**

**Jokes**
- Jokes
  - Jokes

**Non-jokes**
- Non-jokes
  - Non-jokes
Machine learning for pun detection

- Sentences containing puns
- Sentences not containing puns
- Sentences with and without puns

Feature extraction

Learning algorithm

*Training data*

*Test data*

Classifier

Sentences containing puns

Sentences not containing puns
Machine learning for pun detection

Training data

- Sentences containing puns
- Sentences not containing puns

Feature extraction

Learning algorithm

Test data

- Sentences with and without puns

Feature extraction

Classifier

- Sentences containing puns
- Sentences not containing puns
Pun location

- Task: Given some text known to contain a pun, which part is the pun?
- So far only very cursory investigations
- “Highest polysemy” baseline achieves 18% accuracy, compared to 14% for random guessing (Miller, 2016)
- Machine learning approaches might also work here
Pun interpretation

- Task: Given a context containing a pun, and the location of the pun, identify the meaning of the pun and its target
- Prerequisite for imperfect puns: Determine the form of the target
Polysemy is a characteristic of all natural languages.
Polysemy is a characteristic of all natural languages.

“He hit the ball with the bat.”
Polysemy is a characteristic of all natural languages.

“He hit the ball with the bat.”
Polysemy is a characteristic of all natural languages.

“He hit the ball with the bat.”
Polysemy is a characteristic of all natural languages.

“He hit the ball with the bat.”
Polysemy is a characteristic of all natural languages.

“He hit the ball with the bat.”

Word sense disambiguation (WSD) is the task of determining which of a word’s senses is intended in a given context.
Motivation for WSD

Machine translation does not work unless word senses can be disambiguated:

- **English:** bat
- **Romanian:** bâta, liliac, şa
Supervised word sense disambiguation

- Feature extraction
- Sentences using “bat” (club)
- Sentences using “bat” (animal)
- Sentences using “bat” (saddle)
- Sentences using “bat” (unknown)

- Training data
- Test data

- Feature extraction
- Learning algorithm

- Classifier
- Sentences using “bat” (club)
- Sentences using “bat” (animal)
- Sentences using “bat” (saddle)
Knowledge-based WSD relies only on pre-existing, general-purpose linguistic resources such as dictionaries and thesauri. No manually annotated training data is required. More easily applicable and adaptable, but accuracy can be low.
Knowledge-based WSD relies only on pre-existing, general-purpose linguistic resources such as dictionaries and thesauri

- No manually annotated training data is required
- More easily applicable and adaptable, but accuracy can be low
- **Simplified Lesk**: a knowledge-based WSD that uses overlap between context and dictionary definitions
Knowledge-based word sense disambiguation

- Knowledge-based WSD relies only on pre-existing, general-purpose linguistic resources such as dictionaries and thesauri
- No manually annotated training data is required
- More easily applicable and adaptable, but accuracy can be low
- **Simplified Lesk**: a knowledge-based WSD that uses overlap between context and dictionary definitions

“He hit the **ball** with the **bat**.”

- **bat**
  1. A small, nocturnal flying mammal of order *Chiroptera*.
  2. A wooden club used to **hit a ball** in various sports.
  3. A pack saddle.
Knowledge-based word sense disambiguation

- Knowledge-based WSD relies only on pre-existing, general-purpose linguistic resources such as dictionaries and thesauri
- No manually annotated training data is required
- More easily applicable and adaptable, but accuracy can be low
- **Simplified Lesk**: a knowledge-based WSD that uses overlap between context and dictionary definitions

“He **hit** the **ball** with the **bat**.”

<table>
<thead>
<tr>
<th>bat</th>
<th>1. A small, nocturnal flying mammal of order <em>Chiroptera</em>.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2. A wooden club used to <strong>hit</strong> a <strong>ball</strong> in various sports.</td>
</tr>
<tr>
<td></td>
<td>3. A pack saddle.</td>
</tr>
</tbody>
</table>

7 July 2016 | Ubiquitous Knowledge Processing Lab | Department of Computer Science | Tristan Miller | 20
Adapting WSD to (perfect) pun interpretation: Supervised pun interpretation ( naïve)
Challenges to supervised pun interpretation

Knowledge acquisition bottleneck:

- Supervised WSD generally requires a large number of training examples per word sense
- Unrealistic to find large numbers of training examples for each pun
Challenges to supervised pun interpretation

Knowledge acquisition bottleneck:

- Supervised WSD generally requires a large number of training examples per word sense
- Unrealistic to find large numbers of training examples for each pun
- Combinatorial explosion in number of sense combinations:
  - Assuming a perfect pun on a word with \( n \) senses, there are \( \binom{n}{2} = \frac{n!}{2(n-2)!} \) classes to distinguish
  - Number of classes practically limitless for imperfect puns
Adapting WSD for perfect pun interpretation: A slightly less naïve way

- Basic adaptation of WSD systems to pun interpretation:
  - select the two top-scoring senses

- Advantages:
  - straightforward
  - works with both supervised and knowledge-based approaches
Adapting WSD for perfect pun interpretation: A slightly less naïve way

- Basic adaptation of WSD systems to pun interpretation:
  - select the *two* top-scoring senses
- Advantages:
  - straightforward
  - works with both supervised and knowledge-based approaches
- Disadvantages:
  - works only for homographic puns
  - works only for monolexemic puns
Adapting WSD for perfect pun interpretation: Further refinements

- Problem Dictionary sense distinctions often too fine-grained
Adapting WSD for perfect pun interpretation: Further refinements

- Problem Dictionary sense distinctions often too fine-grained
- Work-around: Cluster senses by similarity; ensure that the system does not choose two senses in the same cluster
Example: Using sense clustering to break ties

Where do otters keep their money? At the bank!
Example: Using sense clustering to break ties

Where do otters keep their money? At the bank!

Senses

- sloping land (especially the slope beside a body of water)
- a long ridge or pile
- an arrangement of similar objects in a row or in tiers
- a financial institution that accepts deposits...
- a building in which the business of banking transacted
- a flight maneuver; aircraft tips laterally about its longitudinal axis
Example: Using sense clustering to break ties

Where do otters keep their money? At the **bank**!

<table>
<thead>
<tr>
<th>Scores</th>
<th>Senses</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>sloping land (especially the slope beside a body of water)</td>
</tr>
<tr>
<td>2</td>
<td>a long ridge or pile</td>
</tr>
<tr>
<td>1</td>
<td>an arrangement of similar objects in a row or in tiers</td>
</tr>
<tr>
<td>7</td>
<td>a financial institution that accepts deposits...</td>
</tr>
<tr>
<td>5</td>
<td>a building in which the business of banking transacted</td>
</tr>
<tr>
<td>0</td>
<td>a flight maneuver; aircraft tips laterally about its longitudinal axis</td>
</tr>
</tbody>
</table>
Example: Using sense clustering to break ties

Where do otters keep their money? At the **bank**!

<table>
<thead>
<tr>
<th>Scores</th>
<th>Senses</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>sloping land (especially the slope beside a body of water)</td>
</tr>
<tr>
<td>2</td>
<td>a long ridge or pile</td>
</tr>
<tr>
<td>1</td>
<td>an arrangement of similar objects in a row or in tiers</td>
</tr>
<tr>
<td>7</td>
<td>a financial institution that accepts deposits...</td>
</tr>
<tr>
<td>5</td>
<td>a building in which the business of banking transacted</td>
</tr>
<tr>
<td>0</td>
<td>a flight maneuver; aircraft tips laterally about its longitudinal axis</td>
</tr>
</tbody>
</table>
## Results

<table>
<thead>
<tr>
<th>System</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Lesk-like disambiguator</td>
<td>11.90</td>
</tr>
<tr>
<td>... with sense cluster filter</td>
<td>16.77</td>
</tr>
<tr>
<td>Random baseline</td>
<td>9.31</td>
</tr>
</tbody>
</table>
Adapting WSD for imperfect pun interpretation: Sound similarity
Adapting WSD for imperfect pun interpretation: Sound similarity

- Any pair of words can be characterized by their (perceived) similarity in terms of sound or pronunciation.
- Studying pairs with a phonologically constrained relationship can help us model that relationship.
- Conversely, a model that quantifies perceived sound differences between words can assess the probability of a given relationship.
- In particular, a model of sound similarity could help detect and interpret puns.
Early similarity models

- “Predicted phonetic distance” or “PPD” (Vitz & Winkler, 1973)
  1. Optimally align two phonemic sequences
  2. Compute the relative Hamming distance (i.e., the proportion of non-matching phoneme positions)
Early similarity models

“Predicted phonetic distance” or “PPD” (Vitz & Winkler, 1973)

1. Optimally align two phonemic sequences
2. Compute the relative Hamming distance (i.e., the proportion of non-matching phoneme positions)

```
# Ø Ø Ø Ø Ø r e l e f n # relation
# ^ n d e r i l Ø Ø t n # underwritten
```

$$\text{PPD} = \frac{9}{11} \approx 0.818$$
Early similarity models

- “Predicted phonetic distance” or “PPD” (Vitz & Winkler, 1973)
  1. Optimally align two phonemic sequences
  2. Compute the relative Hamming distance (i.e., the proportion of non-matching phoneme positions)

\[
\begin{align*}
\text{relation} & \quad \text{underwritten} \\
\text{relation} & \quad \text{underwritten}
\end{align*}
\]

\[
\text{PPD} = \frac{9}{11} \approx 0.818
\]

- Method works better when it is applied separately to the syllable onset, nucleus, and coda.
Early similarity models

- “Predicted phonetic distance” or “PPD” (Vitz & Winkler, 1973)
  1. Optimally align two phonemic sequences
  2. Compute the relative Hamming distance (i.e., the proportion of non-matching phoneme positions)

\[
\begin{align*}
\text{relation} & \quad \text{underwritten} \\
\text{relation} & = \frac{9}{11} \approx 0.818
\end{align*}
\]

- Method works better when it is applied separately to the syllable onset, nucleus, and coda.
- Aligning the sequences is a nontrivial task.
Many models compute similarity in terms of the classic feature matrix (Chomsky & Halle, 1968).
Sound similarity based on phonemic features

- Many models compute similarity in terms of the classic feature matrix (Chomsky & Halle, 1968).
- These models often fail to account for many common cases.
Many models compute similarity in terms of the classic feature matrix (Chomsky & Halle, 1968).

These models often fail to account for many common cases. 

*Trying to preserve his savoir faire in a new restaurant, the guest looked down at the eggs the waiter had spilled in his lap and said brightly, “Well, I guess the yolk’s on me!”*
Sound similarity based on phonemic features

- Many models compute similarity in terms of the classic feature matrix (Chomsky & Halle, 1968).
- These models often fail to account for many common cases.

  *Trying to preserve his savoir faire in a new restaurant, the guest looked down at the eggs the waiter had spilled in his lap and said brightly, “Well, I guess the yolk’s on me!”*

- Variously mitigated by the use of multivalued features (Ladefoged, 1995), feature salience coefficients (Kondrak, 2002), and Optimality Theory (Lutz & Greene, 2003).
Similarity models based on puns

- Hausmann (1974) observed an absolute phonemic distance of no more than four
Similarity models based on puns

- Hausmann (1974) observed an absolute phonemic distance of no more than four.
- Lagerquist (1980): puns tend not to insert or delete syllables, nor to change syllable stress; sound changes tend to occur on the stressed syllable.
Similarity models based on puns

- Hausmann (1974) observed an absolute phonemic distance of no more than four
- Lagerquist (1980): puns tend not to insert or delete syllables, nor to change syllable stress; sound changes tend to occur on the stressed syllable
- Zwicky & Zwicky (1986): certain segments do not appear equally often in puns and targets: Y “ousts” X when Y appears as a pun substitute for the latent target X significantly more often than the reverse.
Similarity models based on puns

- Hausmann (1974) observed an absolute phonemic distance of no more than four
- Lagerquist (1980): puns tend not to insert or delete syllables, nor to change syllable stress; sound changes tend to occur on the stressed syllable
- Zwicky & Zwicky (1986): certain segments do not appear equally often in puns and targets: Y “ousts” X when Y appears as a pun substitute for the latent target X significantly more often than the reverse.
- Sobkowiak (1991): pun understandability is maximized when the consonantal skeleton is kept largely intact
Computational pun target recovery

- Past phonological analyses tend to agree
Computational pun target recovery

- Past phonological analyses tend to agree
- Hempelmann (2003) modelled Sobkowiak’s data into a cost function
Computational pun target recovery

- Past phonological analyses tend to agree
- Hempelmann (2003) modelled Sobkowiak’s data into a cost function
- Jaech et al. (2016) compare Hempelmann’s model to one levering automatically learned phone edit probabilities:
Past phonological analyses tend to agree

Hempelmann (2003) modelled Sobkowiak’s data into a cost function

Jaech et al. (2016) compare Hempelmann’s model to one leveraging automatically learned phone edit probabilities:

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Perfect</td>
</tr>
<tr>
<td>Hempelmann</td>
<td>47.8</td>
</tr>
<tr>
<td>Jaech et al.</td>
<td>73.9</td>
</tr>
</tbody>
</table>
Conclusions and future directions

- Pun interpretation is a hard problem
- Machine learning can aid in target recovery for imperfect puns
- Little or no prior work in pun detection and location
- Existing work not deeply based on theories of humour
SemEval-2017 Shared Task on Detection and Interpretation of English Puns

- SemEval: An organized evaluation competition for tasks in computational semantics, since 1998
- Basic shared task setup:
  1. Organizers provide data (annotations withheld)
  2. Participants build annotation systems, submit results
  3. Organizers evaluate, tabulate, and analyze results
  4. Participants write papers describing their systems
- SemEval-2017 to include tasks in pun detection, location, and interpretation
- Two tracks for each task: homographic and heterographic
- Organizers: Iryna Gurevych, Christian F. Hempelmann, Tristan Miller
References and further reading I


References and further reading II


Image credits:

- Woman and laptop ©2012 Shopware. CC BY-SA 3.0.
- Firefox OS Emojis ©2015 Mozilla Foundation. CC BY 4.0.