Semantic Text Mining

1: Fun Facts: Automatic Trivia Fact Extraction from Wikipedia
2: Argument Mining with Structured SVMs and RNNs

Nils Nommensen

17.07.2018
Trivia Extraction: Introduction and Motivation

- Automatically find interesting trivia facts
- Examples:
  - ”Some cats are allergic to humans.”
  - ”Guinness Brewery estimates that 93,000 liters of beer are lost in facial hair each year in the UK alone.”
- Use Wikipedia category system
  - e.g. Barack Obama article is part of categories ”Presidents of the United States”, ”1961 births”, and more
  - Resulting trivia fact: ”Barack Obama is in the group of Presidents of the United States”
- Augment search queries of named entities with trivia
  - Support explorative information gathering process
Trivia Extraction: Problem Formulation

- When is a fact considered trivia-worthy?
- Naive approach: Look at number of articles in category
- Trivia facts need to be surprising, interesting and not obscure
- Define two formal criteria characterizing good trivia
  1. surprise
  2. cohesiveness
Trivia Extraction: Property 1 - Surprise

- **Surprise**: How unusual is it for an article to belong to a category?
- **Similarity between articles**: $\sigma(a, a')$
- **Similarity between article and category**:
  \[
  \sigma(a, C) = \frac{1}{|C| - 1} \sum_{a \neq a' \in C} \sigma(a, a')
  \]
- **Surprise metric**: $\text{surp}(a, C) = \frac{1}{\sigma(a, C)}$
- **Top results from 1940s Hollywood film actress Hedy Lamarr**:
  - ”20th-century Austrian people”
  - ”American anti-fascists”
  - ”Radio pioneers”
Trivia Extraction: Property 2 - Cohesiveness

- Surprise not enough: cohesiveness of group important
- Cohesiveness of category: average similarity between pairs of articles

\[
\text{cohesive}(C') = \frac{1}{\binom{|C'|}{2}} \sum_{a \neq a' \in C'} \sigma(a, a')
\]

- Top articles by cohesiveness:
  - "Metro-Goldwyn-Mayer contract players"
  - "Actresses from Vienna"
  - "Austrian film actresses"
Trivia Extraction: Trivia-Worthiness

- Trivia-worthy, if high in both surprise and cohesive metric
- trivia\((a, C)\) = cohesive\((C)\) × surp\((a, C)\) = \frac{\text{cohesive}(C)}{\sigma(a,C)}
Trivia Extraction: Article Similarity

- Problems with "usual" similarity methods:
  1. Looking for broad similarity
  2. Loss of semantic similarities
- Tackle these problems:
  1. Only use $K$ most important words by computing TF-IDF score
  2. Compute word similarity $\sigma(w_1, w_2)$ using word2vec
- Article similarity for two articles $a_1$ and $a_2$ with top-K TF-IDF terms $T_1$ and $T_2$:
  - For each term in $T_1$ find most similar term in $T_2$ and compute similarity $\sigma(w_1, w_2)$
  - Vice versa for terms in $T_2$
  - Sum up all weights with weighted terms
  - Compute article similarity $\sigma(a_1, a_2)$ by normalizing sum
Trivia Extraction: Evaluation - Crowd-sourcing Study

- Crowd-workers were asked to agree/disagree with the statement:
  - "This is a good trivia fact"
Trivia Extraction: Evaluation - Crowd-sourcing Study

- Crowd-workers were asked to agree/disagree with statement:
  - "This fact is surprising"
Trivia Extraction: Evaluation - Crowd-sourcing Study

- Crowd-workers were asked to agree/disagree with statement:
  - "I knew this fact before reading it here"

**Majority opinion of the statement "I knew this fact before reading it here"**

- **Top**: Yes - 11.01%, No - 95.41%, No majority reached - 3.58%
- **WTM**: Yes - 4.59%, No - 95.41%, No majority reached - 0%
- **Middle**: Yes - 55.05%, No - 44.95%, No majority reached - 0%
- **Bottom**: Yes - 31.19%, No - 66.97%, No majority reached - 1.84%
Trivia Extraction: Evaluation - Ad Engagement Study

- Creating online ads with facts leading to modified Wikipedia page
- Measuring click-through rate [CLR], bounce rate and average dwell time
  - CLR of 0.8% indicated willingness of user to explore trivia facts
  - Bounce rate: bottom trivia(52%) WTM(47%), top trivia(37%)
  - Dwell time: bottom trivia(30s) WTM(43s), top trivia(48s)
Trivia Extraction: Discussion and future work

- Works good for human entities, not so well in other domains
- Limited to scheme 'X is member of group Y'
- Turning facts into questions
- Other Applications
  - Finding false information, Help building educational agents, ...
  - Method of breaking filter bubble
Trivia Extraction: Conclusion

- Time to get away from search engine as passive librarian
- Use of Wikipedia category system
- Proposed algorithm to find trivia-worthy facts of entities
  - Based on surprise and cohesiveness properties
- Crowdsourcing and Ad engagement study
Argument Mining: Introduction

- Identification of argumentative structures in documents
- Example: "People with gene X are at risk of getting cancer, as study Z suggests."
- Argument mining task:
  - Classify propositions
  - Detect argumentative relations
- Previous approaches: Minimum Spanning Tree based algorithms
  - Real world arguments more complicated
- Argument mining as inference on factor graph
- Model parametrization by structured SVM or RNN
Argument Mining: Data

- Possible proposition types: policy, value, fact, ...
- Possible link types: reason, evidence, ...
- Two datasets: CPCD and UKP
- Links may form unrestricted directed graphs

[Calling a debtor at work is counter-intuitive; ]\(_a\)  
[if collectors are continuously calling someone at work, other employees may report it to the debtor’s supervisor. ]\(_b\)  
[Most companies have established rules about receiving or making personal calls during working hours. ]\(_c\)  
[If a collector or creditor calls a debtor on his/her cell phone and is informed that the debtor is at work, the call should be terminated. ]\(_d\)  
[No calls to employers should be allowed, ]\(_e\)  
[as this jeopardizes the debtor’s job. ]\(_f\)
Argument Mining: Factor Graph

- Factor graph: probabilistic graphical model
- Model complex joint probability distributions
- Two possible node types
  - random variables
  - factors
- Advantage: Effective algorithms for approximate inference
- Example:

\[
p(A, B, C, D, E) = \frac{1}{Z} p_1(A, C)p_2(B, C, D)p_3(C, D, E)
\]
Argument Mining: Model description

- Document consists of:
  - Propositions \( \{a, b, c, \ldots\} \)
  - Possible directed links \( a \rightarrow b \)
- We want to predict:
  - Type of each preposition \( y_a \in P = \{\text{Policy, Fact, \ldots}\} \)
  - Binary label for each link \( y_{a \rightarrow b} \in R = \{\text{on, off}\} \)
Argument Mining: Model description

- Random variables (white circles)
  - Random variable for each proposition (a, b, c, ...)
  - Random variable for each possible link (a → b, a → c, ...)

![Diagram showing random variables and links between propositions a, b, and c.]
Argument Mining: Model description

- Unary Potentials (grey boxes)
  - Parametrize unary factors with unary potentials
  - Score of $a$ for each possible proposition type: $\phi(a) \in \mathbb{R}^{|P|}$
  - Score for $a \rightarrow b$ being on or off: $\phi(a \rightarrow b)$
Argument Mining: Model description

- Compatibility factors (black boxes)
  - Dense factor bounding \((a, b, a \rightarrow b)\), scoring their assignment
Argument Mining: Model description

- Second order factors (orange boxes)
  - grandparent \((a \rightarrow b \rightarrow c)\)
  - sibling \((a \leftarrow b \rightarrow c)\)
  - co-parent \((a \rightarrow c \leftarrow b)\)
Argument Mining: Model description

▶ Structural constraints (empty grey boxes)
  ▶ For example enforcing transitivity
  ▶ $a \rightarrow b \land b \rightarrow c \Rightarrow a \rightarrow c$
Argument Mining: Argument Structure SVM

- Use of linear structured support vector machine
- Handcrafted feature vectors:
  - Proposition features: bag of words, presence of connectors, ..., 7100 dimensions
  - Link features: nouns in common, distance, ..., 2100 dimensions
  - Second order features: nouns in common, ..., 35 dimensions
- Optimize for average between link and proposition $F_1$ scores
Argument Mining: Argument Structure RNN

- Encode contextual information using deep bidirectional LSTM model
- Each proposition represented by average of LSTM outputs
- Calculation of the score:
  - Unary proposition potentials $\phi(a)$: MLP
  - Link potentials $\phi(a \rightarrow b)$: MLP with bilinear transformation and 2D weight matrix
  - Second order potentials $\phi(a \rightarrow b \rightarrow c)$: analog with 3D weight tensor
Argument Mining: Results

- Overall trend: Training using structured objective is better
- SVM outperforms RNNs in most settings
- RNNs shine in proposition classification
- SVMs overpredict links, while RNNs underpredict them
Argument Mining: Conclusion

- Argumentation parsing model based on factor graphs
- Parametrized with higher-order factors and link structure constraints
- Linear structured SVMs and RNNs
- SVMs mostly outperform RNNs
- New state of the art results