A Generate and Rank Approach to Sentence Paraphrasing

Prodromos Malakasiotis*
Ion Androutsopoulos*†

* NLP Group, Department of Informatics, Athens University of Economics and Business, Greece
†Digital Curation Unit – IMIS, Research Centre “Athena”, Greece
Paraphrases

• Phrases, sentences, or longer expressions, or patterns with the same or very similar meanings.
  – “X is the writer of Y” ≈ “X wrote Y” ≈ “X is the author of Y”.
  – Can be seen as bidirectional textual entailment.

• Paraphrase recognition:
  – Decide if two given expressions are paraphrases.

• Paraphrase extraction:
  – Extract pairs of paraphrases (or patterns) from a corpus.
  – Paraphrasing rules (“X is the writer of Y” ↔ “X wrote Y”).

• Paraphrase generation (this paper):
  – Generate paraphrases of a given phrase or sentence.
Generate-and-rank with rules

Paraphrasing rules rewrite the source in different ways producing candidate paraphrases.

We focus mostly on the ranker. (We use an existing collection of rules.)

State of the art paraphraser we compare against.

Multi-pivot approach (Zhao et al. ’10)

Pick the candidate(s) with the smallest sum(s) of distances from all other candidates and S.

3 MT engines, 6 pivot languages.
Applying paraphrasing rules

\[ R_1: \text{a lot of } \text{NN}_1 \leftrightarrow \text{plenty of } \text{NN}_1 \]

\[ S_1: \text{He had a lot of admiration for his job.} \]

\[ C_{11}: \text{He had plenty of admiration for his job.} \]

• We use approx. \textbf{1,000,000 existing paraphrasing rules} extracted from parallel corpora by Zhao et al. (2009).
  – Each rule has \textbf{3 context-insensitive scores} \((r_1, r_2, r_3)\) indicating how good the rule is in general (see the paper for details).
  – We also use the \textbf{average} \((r_4)\) of the three scores.

• For each source \((S)\), we produce candidates \((C)\) by using the \textbf{20 applicable rules} with the \textbf{highest average scores} \((r_4)\).
  – Multiple rules may apply in parallel to the same \(S\). We allow all possible rule combinations.
Context is important

- Although we apply the rules with the highest context-insensitive scores ($r_4$), the candidates may not be good.
  - The context-insensitive scores are not enough.

- A paraphrasing rule may not be good in all contexts.
  - “X acquired Y” $\leftrightarrow$ “X bought Y” (Szpektor 2008)
    - “IBM acquired Coremetrics” $\approx$ “IBM bought Coremetrics”
    - “My son acquired English quickly” $\neq$ “My son bought English quickly”
  - “X charged Y with” $\leftrightarrow$ “X accused Y of”
    - “The officer charged John with...” $\approx$ “The officer accused John of...”
    - “Mary charged the batteries with...” $\neq$ “Mary accused the batteries of...”
Our publicly available dataset

• Intended to help train and test alternative rankers of generate-and-rank paraphrase generators.

• 75 source sentences (S) from AQUAINT.

• All candidate paraphrases (C) of the 75 sources generated, by applying the rules with the best 20 context-insensitive scores ($r_4$).

• Test data: 13 judges scored (1 – 4 scale) the resulting 1,935 <S, C> pairs in terms of:
  – grammaticality (GR),
  – meaning preservation (MP),
  – overall quality (OQ).

• Training data: another 1,500 <S, C> pairs scored by the first author in the same way (GR, MP, OQ).
Overall quality (OQ) distribution in test data

More than 50% of the candidate paraphrases judged bad, although we apply only the “best” 20 rules with the highest context-insensitive scores ($r_4$). The ranker has an important role to play!

4: perfect
1: totally unacceptable
Can we do better than just using the context-insensitive rule scores?

- In a **first experiment**, we used **only** the judges’ overall quality scores (OQ).
  - Task: **predict the correct class** of each <S, C> pair.

- **Baseline**: classify each <S, C> pair as positive iff the r₄ score of the rule (or the mean r₄ score of the rules) that turned S into C is **greater than t**.
  - The threshold t was tuned on held-out data.

- Against a **MaxEnt classifier** with 151 features.
The 151 features

- **3 language model** features:
  - Language model score of the **source** (S), of the **candidate** (C), and their **difference**.
  - **3-gram LM** trained on ~6.5 million AQUAINT sentences.

- **12 features** for context-insensitive rule scores.
  - 3 for the **highest**, **lowest**, **mean** $r_4$ scores of the rules that turned S to C. Similarly for $r_1$, $r_2$, $r_3$.

- **136 features** of our **recognizer** (Malakasiotis 2009).
  - Multiple **string similarity measures** applied to original <S,C>, stemmed, POS-tags, Soundex... (see the paper).
  - Similarity of **dependency trees**, **length ratio**, **negation**, WordNet **synonyms**, ...
  - Best published results on the **MSR paraphrase recognition corpus** (with full feature set, despite redundancy).

All features normalized in $[-1, +1]$. 
MaxEnt beats the baseline

MaxEnt error rate on unseen instances (candidate paraphrases).

Baseline (threshold on mean r4 scores).

MaxEnt error rate on training instances encountered (sort of lower boundary). Adding training data would not help.
Using an SVR instead of MaxEnt

- Some **judges said** they were **unsure how much** the **OQ scores** should reflect **grammaticality** (GR) or **meaning preservation** (MP).
- And that we should also consider **how different** (DIV, diversity) each candidate paraphrase (C) is from the source (S).
- **Instead of** (classes of) **OQ scores**, we now use:
  \[ y = \lambda_1 \cdot \text{GR} + \lambda_2 \cdot \text{MP} + \lambda_3 \cdot \text{DIV}, \text{with } \lambda_1 + \lambda_2 + \lambda_3 = 1. \]
  as the **correct score** of each \(<S, C>\) pair.
  – **GR** and **MP**: obtained from the **judges**.
  – **DIV**: **automatically** measured as **edit distance** on tokens.
- **SVRs** similar to SVMs, but for **regression**. Trained on examples \(\langle \vec{x}, y \rangle\), \(\vec{x}\) is a **feature vector**, and \(y \in \mathbb{R}\) is the **correct score** for \(\vec{x}\).
  – In our case, each \(\vec{x}\) represents an **<S, C> pair**.
  – The SVR **tries to guess the correct score** \(y\) of the **<S, C> pair**.
  – RBF kernel, **same features** as in MaxEnt.
Which values of $\lambda_1$, $\lambda_2$, $\lambda_3$?

- By **changing** the **values of** $\lambda_1$, $\lambda_2$, $\lambda_3$, we can force our system to assign **more/less importance** to grammaticality, meaning preservation, diversity.
  - E.g., in **query expansion** for IR, **diversity** may be more **important** than grammaticality and (to some extent) meaning preservation.
  - In **NLG**, **grammaticality** is much more **important**.
  - The $\lambda_1$, $\lambda_2$, $\lambda_3$ values **depend** on the **application**.

- A **ranker dominates** another one iff it performs better for all combinations of $\lambda_1$, $\lambda_2$, $\lambda_3$ values, i.e., in all applications.
  - Similar to comparing **precision/recall** or **ROC curves** in text classification.
How well a ranker predicts the correct $y$ scores.

SVR-REC ranker (151 features): also uses our recognizer’s features.

SVR-BASE (15 features): LM features, features for context-insensitive rule scores.

When $\lambda_3$ is very high, we care only about diversity, and SVR-REC includes features measuring diversity.

$$\lambda_1 + \lambda_2 + \lambda_3 = 1$$
Comparing to the state of the art

• We finally compared our system (with SVR-REC) against Zhao et al.’s (2010) multi-pivot approach.
  – Multi-pivot approach re-implemented.

• The multi-pivot system always generates paraphrases.
  – Vast resources (3 commercial MT engines, 6 pivot languages).

• Our system often generates no candidates.
  – No paraphrasing rule applies to ~40% of the sentences in the NYT part of AQUAINT.

• But how good are the paraphrases, when both systems produce at least one paraphrase?
  – Simulating the case where more rules have been added to our system, to the extent that a rule always applies.
Comparing to the state of the art

300 new source sentences \( (S) \) to which at least one rule applied:

- Top-ranked paraphrase \( (C_1) \) of our system \( (\lambda_1 = \lambda_2 = \lambda_3 = 1/3) \).
- Top-ranked paraphrase \( (C_2) \) of multi-pivot system (ZHAO-ENG).
- Asked 10 judges to score the \( <S, C_1>, <S, C_2> \) for GR and MP; DIV measured automatically as edit distance.

* statistical significance

Our system (with SVR-REC).
Multi-pivot system.
Conclusions

• A new **generate-and-rank** method to **paraphrase** sentences.
  
  – Existing **paraphrasing rules** generate candidate paraphrases, and an **SVR ranker** (or MaxEnt) selects the best.
  – Can be **tuned** to assign **more/less importance** to grammaticality, meaning preservation, diversity
  – Performs well against state-of-the-art multi-pivot paraphraser, when paraphrasing rules apply.

• A new **methodology** and **publicly available dataset** to **evaluate different ranking components** of generate-and-rank paraphrasers.
  
  – Across **different combinations of weights** for grammaticality, meaning preservation, diversity.
Future work

• **Compare** to the **multi-pivot** approach for **more combinations** of $\lambda_1$, $\lambda_2$, $\lambda_3$ values.
  – Instead of only $\lambda_1 = \lambda_2 = \lambda_3 = 1/3$.

• **Add more paraphrasing rules.**
  – To be able to **paraphrase more source sentences**.

• **Combine** the **multi-pivot** approach and our **SVR ranker**.
  – **Generate candidates with both paraphrasing rules** and as in the **multi-pivot approach**.
  – **Rank them with (a version of) our SVR ranker**.

• **Use paraphrase generation in larger systems (IR, QA, NLG)** and in **sentence compression**.
  – See our **UCNLG+Eval paper** on sentence compression.