

Meaningful Term Extraction and Discriminative Term Selection in Text Categorization via Unknown-Word Methodology

Yu-Sheng Lai and Chung-Hsien Wu
(National Cheng Kung University, Taiwan)

Presented by Graeme Hirst

ACM Transactions on Asian Language Information Processing, 1(1), March 2002, 34–64.
Also based on the authors' "Unknown word and phrase extraction using a phrase-like-unit-based likelihood ratio", *International Journal of Computer Processing of Oriental Languages*, 13(1), March 2000, 83–95.

1



Yu-Sheng Lai and Chung-Hsien Wu

2

Text classification

- Assigning each of a set of texts to one or more pre-defined categories according to its content.
 - ▶ Routing customer-service or technical-support requests.
 - ▶ Filtering spam e-mail, offensive Web sites, etc.
 - ▶ Classifying news articles by broad category or narrow topic.
 - ▶ ...
- Many different approaches: rule-based, statistical; manual, machine-learning.
- Relies on looking at certain **features** of the text.

3

Features

- Represent document as a **vector** (length n) of n boolean or numerical features.
- Surface form of text:
 - ▶ *e.g.*, > 10% upper-case \Rightarrow spam.
- Words and phrases in text:
 - ▶ *e.g.*, *click*, *per*, *ff0000*, *teens*, *Nigerian* \Rightarrow spam.*
 - ▶ *e.g.*, *though*, *apparently*, *indehiscent* \nRightarrow spam.*
- Can give different **weights** to different features.

*Some examples are from: Paul Graham, "A plan for spam", August 2002.

4

Discriminative power of features

- Want to find features that truly discriminate between categories.
 - ▶ *e.g., fast, make, lose* $\stackrel{?}{\Rightarrow}$ spam.
- **Training data:** Set of already-classified data; use it to determine a mapping from features to categories.
 - ▶ Ideally, the learning algorithm will identify which features are discriminative and which aren't.

5

Vectors of features, bags of words

- Canonical approach: Consider document to be a **bag of words**; features are the frequency of each word-type.
- *Problem:* Unlimited number of word-types.
Solution: Use only the n most common words in the pre-defined **lexicon**.
- *Problem:* Very common or general words (*the, have, day*) aren't discriminative.
Solution: Exclude them (**stop list**).

6

Limitations of this approach

- Have thrown away all words, names that aren't in lexicon; but they are discriminative.
 - ▶ *Britney* ⇒ entertainment; *indehiscent* ⇒ botany.
- Have thrown away information on word-order, phrases, etc.; but this is discriminative.
 - ▶ *lose, weight, fast* $\stackrel{?}{\Rightarrow}$ spam.
lose weight fast ⇒ spam.
- *Solution*: Don't. Keep and use this information.
- *A fortiori*: Use *only* this information as the features for classification.

7

Chinese text

- Basic character is morpheme; several thousand in set.
- Characters carry meaning, not sound.
- Words are from one to four (or more) characters long.
- Text is written without breaks between words.
 - ▶ But many one- and two-character strings are words.
 - ▶ Hence high degree of local ambiguity in text.

Acknowledgements: Yun Niu; *Encyclopedia of Language and Linguistics*.

8

“Phase-like units” (1)

- Find “Phase-like units” in the corpus.
 - ▶ *i.e.*, **collocations**. Previous research not acknowledged.
- Unclear from paper whether basic units are morphemes / characters or words.
 - ▶ Earlier paper: First use lexicon to segment text into known words. But mightn’t that obscure the unknown words?
 - ▶ Example: Input text: ... *quicksandstorm*...
Lexicon: ..., *quick*, *sand*, *sandstorm*, *storm*, ...
Unknown word *quicksand* cannot be discovered if *sandstorm* is already picked out.

9

“Phase-like units” (2)

- This paper: Lexicon hardly mentioned, but basic units are “words”.
 - ▶ “Unknown words and phrases are always the combination of some words defined in the lexicon and some ~~unknown~~ <additional> characters.”

10

Phrase likelihood ratio (PLR)

- If a string p of $n > 1$ words w_1, \dots, w_n can be broken into two substrings w_1, \dots, w_i and w_{i+1}, \dots, w_n such that the frequency of the whole string is almost as great as that of one of the substrings, then s is probably a phrase.
- $PLR(p) = \max_i \frac{tf(p)}{\min\{tf(w_1 \dots w_i), tf(w_{i+1} \dots w_n)\}}$
- Accept as PLU if $PLR \geq 1 - \epsilon$. Require at least c occurrences of p ; and the greater its frequency, the lower the threshold of acceptance.

11

Example

勇士隊總教練杜福明表示

The chief coach of Braves, Du Fu-Ming, said

12

“Purification”

- 1. Discard stop terms: addresses, dates, quantities, etc, as identified by presence of certain words.
- 2. Discard PLUs that are substrings of other PLUs whose frequency is equal and whose PLR is at least as good.
- 3. For other cases where some PLUs are substrings of others, compute a “reliability” for each by comparing its tf and PLR to those of its contained and containing substrings.
 - ▶ Score +1 for greater, -1 for less; sum over all comparisons.

Discard those with overall negative reliability.

13

Example

勇士隊總教練杜福明表示

The chief coach of Braves, Du Fu-Ming, said

- We now have a lexicon of PLUs that aren't in the lexicon.
 - ▶ About 81% of them are true meaningful terms.

14

Discriminability (1)

- But which PLUs actually help with text classification?
- Those that, in training corpus, turn up a lot more in one category than the others do.
- A kind of *tf·idf* measure, but without logarithmic damping. Previous research not acknowledged (here).
- Can discard terms whose discriminability is below some threshold (as determined experimentally).
 - ▶ Do this separately for each category.

15

Discriminability (2)

- **Discriminability** of term t for category g :

$$W(t, g) = \frac{p(t \in g)}{\max_{c \neq g} p(t \in c)}$$

where $p(t \in c)$ is the (smoothed, kind-of) probability that a given token in c is t .*

$$p(t \in c) = \frac{tf_c(t) + 1}{|c| + 1}$$

where $|c|$ is the number of tokens in the category c in the training corpus.

*Not the probability, given category c , that t is present somewhere.

16

Representing a document

- Training corpus has docs in K different categories.
- Each document $D_{k,j}$ in category k is represented as a vector length n , whose i th component $c_{k,j,i}$ is the frequency* in the document of PLU number i (t_i), weighted by its PLR and its discriminability for the category:

$$c_{k,j,i} = tf(t_i, D_{k,j}) \cdot PLR(t_i) \cdot S(W(t_i, k))$$

where S is a smoothing function onto $[0,1]$.

- Represent each category k as the mean v_k of all docs in category.

*Smart indexing method to find, index, and count all the PLUs in each document in the collection by compiling an FSA from the PLU lexicon.

17

Classifying a document

- When a new doc X is to be classified, represent it the same way for each of the K possible categories.
 - ▶ A vector of K vectors.
(An indexed set X of vectors x_1, \dots, x_K would suffice.)
- To classify X , get the cosine distance between x_k and v_k for each category k .
 - ▶ The biggest one wins.
(Exactly one category per doc.)

18

Experiments (1)

- **Corpus:** 44,000 news articles (35M words), classified into five upper and 36 lower categories; about 90% used for training, 10% for testing.
- Approx 85,000 raw PLUs derived from corpus, purified down to 20,000.
- **Baseline:** Instead of PLUs, use lexicon of 50K(!) words (“purified” from 84K) and *tf·idf* weights.
 - ▶ **Results:** 80% accuracy at first layer, 75% at second layer (72.5% to 76.5%).
 - ▶ Also tried it with discriminability measure instead of *tf·idf*; bad results: from 28% to 99%, average 71%.

19

Experiments (2)

- Try different thresholds for discriminability (with or without a normalization function), and also try constraining each category to use the same number of PLUs (the minimum number that discriminability selects for any category).
 - ▶ **Results:** First-layer accuracy is 93.5% to 95.3%, second-layer accuracy is 85% (75% to 90%); problem of sparse data at second layer.

20

Experiments (3)

- Try different levels of purification (none, stop words, full).
 - ▶ **Results:** Purification improves results in most categories, but degrades results in one.
- Problem: Sparse data for second-layer classification with PLUs.

Try somehow combining PLUs and words (union of both sets of features?), and compare discriminability with *tf·idf*.

 - ▶ **Results:** Discriminability did better than *tf·idf*, and at the second layer, the combination did better than PLUs alone (87.4%).

21

Comparison with other methods

- ***k*-nearest neighbours:** The *k* training docs closest to the new doc vote (weighted by distance) on its category. (Implies all docs in same vector space.) Can use words or PLUs.
- **RIPPER:** An algorithm for learning classification rules.
 - ▶ “Context-sensitive”, “non-linear”: A rule may be a conjunction of $word \in doc$ conditions.
- **Sleeping experts:** Context-sensitive, uses word order. Finds words or phrases (possibly with holes) that correlate positively or negatively with categories, which then contribute a weighted vote on a new doc.

22

Results

- *kNN* with PLUs is brilliant: 96.4%. All the others are way behind.

23

Conclusions (1)

- PLUs are quite discriminative. Most occur in only one (first-layer) category.
- Conflation of two ideas:
 - ▶ Unknown words: names, technical terms, etc; may be single word or multiple words.
 - ▶ Phrase-like units collocations of words that *are* in lexicon.
- Role of morphemic writing system (Chinese)?
 - ▶ Latin scripts: Easy (modulo morphology, agglutination) to find both.

24

Conclusions (2)

- Conflation of research on feature selection with other aspects of text classification seems misplaced. In the end, *kNN* was better than the featured ideas.