Fast Logistic Regression for Text Categorization with Variable-Length N-grams

Georgiana Ifrim*, Gökhan Bakır+, Gerhard Weikum*

* Max-Planck Institute for Informatics
  Saarbrücken, Germany

+Google Switzerland GmbH
  Zürich, Switzerland
Outline

- Background
- Related Work
- Our Approach
  - Structured Logistic Regression
- Experiments
- Conclusion
Text Categorization using Machine Learning

- Typical steps: Training docs -> Feature selection -> Build classifier -> Classify test docs
- Typical document representation: bag-of-words
  - Document: “This phone is good.”

\[ x = (..., \text{is, phone, good, this,} ...) \]

\[ \beta = (..., -0.1, 0.3, 0.7, -0.3, ...) \]

- By this representation we lose: structure, order, context.
- Should we care?
Background - Motivation

- **+1** This product is not **bad**, it’s actually quite **good**.
  - **-1** This product is not **good**, it’s actually quite **bad**.

Bag-of-words representation for both samples:
- (actually, **bad**, it’s, **good**, not, product, quite, This)

N-gram representation:
- **+1** (This, product, is, not, bad, it’s, actually, quite, good, This product, product is, **not bad**, …, **quite good**, This product is, product is not, **is not bad**, …, it’s actually quite good, ….)

- **-1** (This, product, is, not, good, it’s, actually, quite, bad, This product, product is, **not good**, …, **quite bad**, This product is, product is not, **is not good**, …, it’s actually quite bad, ….)
Background - Motivation

- Simple solution to this problem: use syntax clues, use n-grams for fixed n (e.g. all bigrams), e.g. feature engineering

- Problem with this idea: feature selection depends on
  - application, language, domain, corpus size

- Ideally:
  - Regard text as a sequence of word or character tokens
  - Consider as features all the possible subsequences (n-grams) in the training set (But: \( u = \# \text{ unigrams}, \ d = O(u^n) \text{ n-grams} \))
Our Contribution

- A new (Structured) Logistic Regression algorithm able to select a compact set of discriminative variable-length n-grams

- SLR exploits the structure of the n-gram space in order to transform the learning problem into a search problem

- SLR learns: …, not bad, quite good, … …, not good, quite bad, …
Related Work

- Efficient, regularized learning algorithms
  - SVM$^{\text{perf}}$ (Joachims06)
  - Sparse logistic regression (Genkin07: BBR, Komarek03: LR-TRIRLS, Efron04, Zhang01)

- Text Categorization using word/char n-grams
  - Markov chain models
    - fixed order (n-gram language models, Peng2004), variable order (PST - Prediction Suffix Trees, Dekel04), (PPM, PPM* - Prediction by Partial Matching, Cleary97)
  - SVM with string kernel (Lodhi01), efficient feature selection + SVM (Zhang06)

- Optimization approaches to solving logistic regression
  - Newton algorithms (O(d$^2$) memory)
  - Limited memory BFGS (O(d)), conjugate gradient (O(d), Nocedal06), iterative scaling (O(d), Jin03), cyclic coordinate descent (O(d), Shevade03, Zhang01)
The Problem

Given:

- X set of samples
- \( D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\} \) training set, \( y_i \in \{0, 1\} \)
- Features: all n-grams in the training set
- \( d = \# \text{ distinct n-grams in training set} \)
- \( \beta = (\beta_1, \beta_2, \ldots, \beta_d) \) parameter vector
- Log-likelihood of training set for logistic regression

\[
\ell(\beta) = \sum_{i=1}^{N} \left[ y_i \cdot \beta^T \cdot x_i - \log (1 + e^{y_i \cdot \beta^T \cdot x_i}) \right]
\]

Goal:

- Learn a mapping \( f : X \rightarrow \{0, 1\} \) from given training set \( D \)
- Solved by: Find parameter vector \( \beta \) that maximizes \( \ell(\beta) \)
Our Approach

- Solve logistic regression by **coordinate-wise gradient ascent**
  \[ \beta^{new} = \beta^{old} + \varepsilon \cdot \frac{\partial l}{\partial \beta}(\beta^{old}) \], \varepsilon = \text{step size (line search)}

- In each iteration: estimate ascent direction and step size.
  Computing full gradient vector not feasible for the space of all possible n-grams in the training set (u = # unigrams, d = O(u^n))
  \[ \frac{\partial l}{\partial \beta}(\beta) = \left( \frac{\partial l}{\partial \beta_1}(\beta), \frac{\partial l}{\partial \beta_2}(\beta), \ldots, \frac{\partial l}{\partial \beta_d}(\beta) \right) \]

- **Our ascent direction:**
  \[ \left( \frac{\partial l}{\partial \beta}(\beta) \right)^{sparse} = \left( 0, 0, \ldots, 0, \frac{\partial l}{\partial \beta_j}(\beta), 0, \ldots, 0, 0 \right), \text{ where } \beta_j = \arg \max_{\beta_j, j \in \{1, d\}} \left| \frac{\partial l}{\partial \beta_j}(\beta) \right| \]
Searching & Pruning

- Goal: Find coordinate $j$ s.t. $\beta_j = \arg \max_{\beta_j, j \in 1,d} \left| \frac{\partial l}{\partial \beta_j} (\beta) \right|

- For super sequence $s_{j'} \supseteq s_j$, find upper bound on gradient value as a function of subsequence $s_j$: $\text{gradient}(s_{j'}) \leq \mu(s_j)$

- Can prune the search space starting at $s_j$ if $\mu(s_j) \leq \delta$, where $\delta$ is the current suboptimal gradient value
Branch-and-bound strategy

If $\mu(\text{aaa}) = 0.72$ then the gradient of any super-sequence of $\text{aaa}$ is not better than 0.72

$$s_j' \supseteq s_j \quad \text{gradient}(s_j') \leq \mu(s_j)$$

Goal: Find coordinate $j$ s.t. 

$$\beta_j = \arg \max_{\beta_j, j \in 1, d} \left| \frac{\partial l}{\partial \beta_j} (\beta) \right|$$
Upper Bound

Theorem:

For any super sequence $s_j' \supseteq s_j$ it holds that

$$\left| \frac{\partial l}{\partial \beta_j} (\beta) \right| \leq \mu(\beta_j)$$

where

$$\mu(\beta_j) = \max \left\{ \sum_{\{i|y_i=1, s_j \in x_i\}} x_{ij} \left( \frac{e^{\beta^T \cdot x_i}}{1 + e^{\beta^T \cdot x_i}} \right), \sum_{\{i|y_i=0, s_j \in x_i\}} x_{ij} \left( \frac{e^{\beta^T \cdot x_i}}{1 + e^{\beta^T \cdot x_i}} \right) \right\}$$
Experiments

Test collections

IMDB: movie genre classification

- Categories: Crime vs Drama; 7,440 documents; 63,623 word unigrams, 800,000 up to trigrams, 1.9 million up to 5-grams
- Task: Given movie plots, learn the genre of movies

CHINESE: topic detection for Chinese text

- Corpus: TREC-5 People’s Daily News (3600 documents, 4,961 characters)
- Categories: (1) Politics, Law and Society; (2) Literature and Arts; (3) Education, Science and Culture; (4) Sports; (5) Theory and Academy; (6) Economics.
Experiments

- **Compared Methods**
  - Linear training time SVM: $\text{SVM}^{\text{perf}}$ (Joachims, KDD06)
  - Bayesian Logistic Regression: $\text{BBR}$ (Genkin et al, Technometrics07)
  - Structured Logistic Regression: $\text{SLR}$ (Ifrim et al, KDD08)

- **Methodology**
  - SVM, BBR: generate space of variable-length n-grams explicitly, e.g. for $n=3$, generate all unigrams, bigrams and trigrams
  - Evaluate all methods w.r.t. training run-times, micro/macro-avg F1
  - Parameters:
    - SVM: $C = 100$ (fixed), $C \in \{100, \ldots, 500\}$ (tuning)
    - BBR: `-p 1 -t 1` (fixed), `-p 1 -t 1 -C 5 -autosearch` (tuning)
    - SLR: number of optimization iterations, set by fixed threshold (0.005) on the aggregated change in score predictions, or varied between 200 and 1,000 for tuning
IMDB – Macro-avg F1

- 5-fold CV; Plots show Macro-avg F1, Micro-avg F1 similar
- \( n = \max \text{n-gram length}, \text{i.e. all n-grams up to length } n \)

- SLR as good as the state-of-the-art in terms of generalization ability

KDD08 Fast Logistic Regression for Text Categorization with Variable-Length N-grams
IMDB – Running time

- 5-fold CV; average running time per CV split averaged over topics
- $n = \text{max n-gram length}$, i.e. all n-grams up to length $n$

SLR more than one order of magnitude faster than both SVM and BBR

KDD08    Fast Logistic Regression for Text Categorization with Variable-Length N-grams
n = max n-gram length, i.e. all n-grams up to length n

SLR as good as the state-of-the-art in terms of generalization ability
Char n-grams avoid the need for word segmentation in Asian text
CHINESE – Running time

- Training running time
- $n = \text{max n-gram length, i.e. all n-grams up to length } n$

Running time comparison (char n-grams)

- SLR more than one order of magnitude faster than both SVM and BBR
## Interpretability – A look at the models

**IMDB: Crime vs Drama**

<table>
<thead>
<tr>
<th>Word n-grams</th>
<th>Char n-grams</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Crime</strong> n-grams</td>
<td>0.0078 gang war</td>
</tr>
<tr>
<td>0.0067 in the middle of</td>
<td>0.039 gang</td>
</tr>
<tr>
<td>0.0058 hired by</td>
<td>0.038 olice</td>
</tr>
<tr>
<td>0.0048 from prison</td>
<td>0.034 crime</td>
</tr>
<tr>
<td><strong>Drama</strong> n-grams</td>
<td>-0.0069 they meet</td>
</tr>
<tr>
<td>-0.0066 group of people</td>
<td>-0.015 otion</td>
</tr>
<tr>
<td>-0.0015 finds himself</td>
<td>-0.014 urney</td>
</tr>
<tr>
<td>-0.0014 an affair with</td>
<td>-0.011 choo</td>
</tr>
</tbody>
</table>

- Char n-gram models select characteristic substrings (implicit stemming, robustness to morphological variations)
Conclusion

- **Structured Logistic Regression**: a new algorithm for efficiently learning a classifier with unbounded n-grams

- No prior feature selection needed

- No restriction on the n-gram length

- Using n-grams for text classification is beneficial and can be done efficiently
Future Work

- Looking at other text categorization applications (spam filtering, e-mail categorization, etc.)
- Applying this model to supervised information extraction
- Carrying the same gradient projection ideas to other learning problems
Thank you!

Open source software for SLR:
http://www.mpi-inf.mpg.de/~ifrim/slr