A fast and simple algorithm for training neural probabilistic language models

Andriy Mnih & Yee Whye Teh
Gatsby Computational Neuroscience Unit
University College London

Overview
• In spite of their superior performance, neural probabilistic language models (NPLMs) are far less widely used than n-gram models due to their notoriously long training times.
• We introduce a simple training algorithm for NPLMs based on noise-contrastive estimation, with time complexity independent of the vocabulary size.
• Over an order of magnitude faster than maximum-likelihood estimation.
• The resulting models perform just as well.
• We demonstrate the algorithm’s scalability by training several large neural language models on the MSR Sentence Completion Challenge dataset, achieving state-of-the-art results.

Neural probabilistic language models
• Neural probabilistic language models use distributed representations of words to deal with the curse of dimensionality.
  – Words are represented with real-valued feature vectors learned from data.
  – A neural network maps contexts (sequences of word feature vectors) to next word distributions.
• NPLMs generalize well because smooth functions map nearby inputs to nearby outputs.
• Similar representations are learned for words with similar usage patterns.
• Main drawback: very long training times.

Statistical language modelling
• Goal: Model the joint distribution of words in a sentence.
• Applications: speech recognition, machine translation, information retrieval.
• Markov assumption:
  – The distribution of the next word depends only on k words that immediately precede it.
  – Though clearly false, the assumption makes the task much more tractable without making it trivial.

n-gram models
• Task: predict the next word w, from n − 1 preceding words h = w₀,...,wₙ₋₁ (called the context).
• n-gram models are conditional probability tables for P(wₙ|h).
  – Estimated by smoothing word n-tuple counts.
  – Most widely used statistical language models due to their simplicity and good performance.
• Cannot take advantage of similarity between words / contexts.
• Curse of dimensionality:
  – The number of model parameters is exponential in the context size.
  – Cannot take advantage of large context sizes.

Maximum-likelihood estimation
• The gradient of the log-likelihood is:
  \[ \frac{\partial}{\partial \theta} \log P_h(\theta) = \frac{\partial}{\partial \theta} \log P_h(w|h) \approx \frac{\partial}{\partial \theta} \log Z_h(\theta) \]
  Computing \( \frac{1}{w} \log Z_h(\theta) \) is expensive – the time complexity is linear in the vocabulary size.
• Can approximate \( \frac{1}{w} \log Z_h(\theta) \) using importance sampling (Bengio and Senécal, 2003):
  – Sample words from a proposal distribution and reweight the gradients.
• Stability issues: need either a lot of samples or an adaptive proposal distribution.

Noise-contrastive estimation
• Idea: Fit a density model by learning to discriminate between samples from the data distribution and samples from a known noise distribution (Gutmann and Hyvärinen, 2010).
• If noise samples are k times more frequent than data samples, the posterior probability that a sample came from the data distribution is:
  \[ P(\theta | D = 1 | w) = \frac{P_h(w)}{P_h(w) + kP_N(w)} \]
• To fit a model \( P_h(\theta) \) to the data, use \( P_N(\theta) \) in place of \( P_h(\theta) \) and maximize \( P(\theta) \):
  \[ E_{w,h} \left[ \log \frac{P_h(w)}{P_h(w) + kP_N(w)} + kE_{w} \left[ \log \frac{kP_h(w)}{P_h(w) + kP_N(w)} \right] \right] \]
• NCE allows working with unnormalized distributions \( P_h(\theta) \).
  – Set \( P_N(\theta) = P_h(\theta)/Z_h \) and learn \( Z_h \).
  – \( \theta \) are the parameters of the unnormalized distribution and \( s \) is \( \{\theta, \log Z_h\} \).
• The gradient of the objective for context h is:
  \[ \frac{\partial}{\partial \theta} P(\theta) = \frac{\partial}{\partial \theta} \log P_h(w|h) \approx \frac{\partial}{\partial \theta} \log Z_h(\theta) \]

Speedup over MLE
The NCE parameter update is \( \frac{\partial}{\partial \theta} J(\theta) \) times faster than the ML update.
• Here \( c \) is the context size, \( d \) is the feature vector dimensionality, \( \gamma \) is the vocabulary size, and \( k \) is the number of noise samples.

Penn Treebank results
Data: news stories from Wall Street Journal
• Training/validation/test set: 930K/74K/82K words
• Vocabulary: 10K words

Sentence completion results
Task: Given a sentence with a missing word find the correct completion from a list of candidate words.
• Training set: 522 19th-century novels (48M words)
• Test set: 1,040 sentences from five Sherlock Holmes novels
• Five candidate completions per sentence.

Conclusions
Noise-contrastive estimation provides a fast and simple way of training neural language models:
• Over an order of magnitude faster than maximum-likelihood estimation.
• Models trained using NCE with 25 noise samples per datapoint perform as well as the ML-trained ones.