Machine Learning in NLP

Joakim Nivre

Uppsala University
Linguistics and Philology
Why do we use machine learning in NLP?
When should we (not) use machine learning?
Why do we use machine learning in NLP?
When should we (not) use machine learning?

Appropriate problems for machine learning (from Lecture 1):
- Problems for which there is no known exact method
- Problems for which the exact method is too expensive
- Problems that evolve over time
Eugene Wigner’s article “The Unreasonable Effectiveness of Mathematics in the Natural Sciences” examines why so much of physics can be neatly explained with simple mathematical formulas such as $f = ma$ or $e = mc^2$. Meanwhile, sciences that involve human beings rather than elementary particles have proven more resistant to elegant mathematics. Economists suffer from physics envy over their inability to neatly model human behavior. An informal, incomplete grammar of the English language runs over 1,700 pages. Perhaps when it comes to natural language processing and related fields, we’re doomed to complex theories that will never have the elegance of physics equations. But if that’s so, we should stop acting as if our goal is to author extremely elegant theories, and instead embrace complexity and make use of the best ally we have: the unreasonable effectiveness of data.

Alon Halevy, Peter Norvig and Fernando Pereira. 2010. The Unreasonable Effectiveness of Data. IEEE Intelligent Systems.
Plan for this Lecture

▶ A historical perspective
  ▶ Formal theory-driven systems
  ▶ Statistical methods
  ▶ Deep learning
▶ Strengths and weaknesses of machine learning in NLP
Running Example: Parsing

- Input: natural language sentence (word sequence)
- Output: tree or graph capturing syntactic structure
Computational Linguistics in the 1980s

PROCESSING ENGLISH WITH A GENERALIZED PHRASE STRUCTURE GRAMMAR

Jean Mark Gawron, Jonathan King, John Lamping, Egon Loebner,
E. Anne Paulson, Geoffrey K. Pullum, Ivan A. Sag, and Thomas Wasow

Computer Research Center
Hewlett Packard Company
1501 Page Mill Road
Palo Alto, CA 94304

ABSTRACT

This paper describes a natural language processing system implemented at Hewlett-Packard’s Computer Research Center. The system’s main components are: a Generalized Phrase Structure Grammar (GPSG); a top-down parser; a logic transducer that outputs a first-order logical representation; and a “disambiguator” that uses sortal information to convert “normal-form” first-order logical expressions into the query language for HIRE, a relational database hosted in the SPHERE system. We argue that theoretical developments in GPSG syntax and in Montague semantics have specific advantages to bring to this domain of computational linguistics. The syntax and semantics of the system are totally domain-independent, and thus, in principle, highly portable. We discuss the prospects for extending domain-independence to the lexical semantics as well, and thus to the logical semantic representations.

...can be achieved without detailed syntactic analysis. There is, of course, a massive pragmatic component to human linguistic interaction. But we hold that pragmatic inference makes use of a logically prior grammatical and semantic analysis. This can be fruitfully modeled and exploited even in the complete absence of any modeling of pragmatic inferencing capability. However, this does not entail an incompatibility between our work and research on modeling discourse organization and conversational interaction directly. Ultimately, a successful language understanding system will require both kinds of research, combining the advantages of precise, grammar-driven analysis of utterance structure and pragmatic inferencing based on discourse structures and knowledge of the world. We stress, however, that our concerns at this stage do not extend beyond the specification of a system that can efficiently extract literal meaning from isolated sentences of arbitrarily complex grammatical structure. Future systems will exploit the literal meaning thus extracted in more...
Computational Linguistics in the 1980s

- Languages described by formal systems
  - Inventory of elementary units (lexicon)
  - Rules for combining units (grammar)
- Created by linguists in a theoretical framework
  - Linguistic levels: morphology, syntax, semantics
  - Generate all and only well-formed expressions
- Combined with algorithms for analysis/synthesis
Issues

- Coverage
  - Hard to build a complete description of a language
  - Languages are constantly changing

- Robustness
  - Language use is not always well-formed
  - Made worse by lack of coverage
Issues

▶ Ambiguity
  ▶ Natural language grammars inherently ambiguous
  ▶ Combinatorial explosion from interacting rules and levels
  ▶ Practical applications need disambiguation
Statistical NLP in the 1990s

Three New Probabilistic Models for Dependency Parsing: An Exploration

Jason M. Eisner
CIS Department, University of Pennsylvania
200 S. 33rd St., Philadelphia, PA 19104-6389, USA
jeisner@linc.cis.upenn.edu

Abstract
After presenting a novel $O(n^3)$ parsing algorithm for dependency grammar, we develop three contrasting ways to stochasticize it. We propose (a) a lexical affinity model where words struggle to modify each other, (b) a sense tagging model where words fluctuate randomly in their selectional preferences, and (c) a generative model where the speaker fleshes out each word’s syntactic and conceptual structure without regard to the implications for the hearer. We also give preliminary empirical results from evaluating the three models’ parsing performance on annotated Wall Street Journal training text (derived from the Penn Treebank). In these results, the generative model performs significantly better than the others, and does about equally well at assigning part-of-speech tags.

Figure 1: (a) A bare-bones dependency parse. Each word points to a single parent, the word it modifies; the head of the sentence points to the EOS (end-of-sentence) mark. Crossing links and cycles are not allowed. (b) Constituent structure and subcategorization may be highlighted by displaying the same dependencies as a lexical tree.
Eisner’s Model C

\[
Pr(\text{words, tags, links}) = \prod_{1 \leq i \leq n} \left( \prod_{c=-(1+\#\text{left-kids}(i)), c \neq 0}^{1+\#\text{right-kids}(i)} Pr(\text{tword(kid}_c(i)) \mid \text{tag}(\text{kid}_{c-1}(i)), \text{tword}(i)) \right)
\]

- Stochastic process generating a dependency tree
  - Tree probability = product of subtree probabilities
  - Subtree probability = product of child probabilities
  - Child conditioned on tagged head word and preceding child tag
Statistical NLP in the 1990s

- Probabilistic models of language
  - Generative models of $P(X, Y)$
  - Examples: HMM, PCFG, NB
- Parameters estimated from (annotated) data
  - Maximum-likelihood estimation
  - Smoothing to cope with sparse data
- Inference algorithms for analysis:
  - Exact argmax search using dynamic programming
  - Examples: Viterbi, CKY
How Does This Help?

- Ambiguity
  - Disambiguation through probability ranking
  - Learning from data more effective than heuristics
  - Statistical evaluation to measure progress
How Does This Help?

- **Coverage**
  - Smoothing allows graceful degradation
  - Unknown words can be interpreted in context

- **Robustness**
  - Probability ranking allows constraint relaxation
  - No sharp line between well-formed and deviant
A New Paradigm

- Emphasis on robust large-scale processing
- Quantitative evaluation
  - Naturally occurring test data
  - Exact numerical metrics (frequency-based)
- Data-driven development
  - Naturally occurring training data
  - Models induced using statistical inference
Machine Learning?

- Statistical models of the (early) 90s:
  - Generative models of $P(X, Y)$
  - Maximum likelihood estimation (with smoothing)
  - No advanced learning algorithms – just counting

- Main limitation:
  - Rigid independence assumptions (local context)
  - Required for effective learning and efficient inference
Online Large-Margin Training of Dependency Parsers

Ryan McDonald  Koby Crammer  Fernando Pereira
Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA
{ryantm,crammer,pereira}@cis.upenn.edu

Abstract

We present an effective training algorithm for linearly-scored dependency parsers that implements online large-margin multi-class training (Crammer and Singer, 2003; Crammer et al., 2003) on top of efficient parsing techniques for dependency trees (Eisner, 1996). The trained parsers achieve a competitive dependency accuracy for both English and Czech with no language specific enhancements.

models of the same vintage even though it scores parsing decisions in isolation and thus may suffer from the label bias problem (Lafferty et al., 2001).

Discriminatively trained parsers that score entire trees for a given sentence have only recently been investigated (Riezler et al., 2002; Clark and Curran, 2004; Collins and Roark, 2004; Taskar et al., 2004). The most likely reason for this is that discriminative training requires repeatedly reparsing the training corpus with the current model to determine the parameter updates that will improve the training criterion. The reparsing cost is already quite high
McDonald’s Discriminative Model

- Discriminative model of trees given sentences
  - Online learning (perceptron style)
  - Max-margin objective (MIRA)
  - Rich features over the input-output space
Conditional or discriminative models
  - Models for prediction $X \rightarrow Y$
  - Examples: Perceptron, SVM, MaxEnt
Parameters estimated from (annotated) data
  - Learning as numerical optimization
  - Regularization to prevent overfitting
Inference algorithms for analysis
  - Exact argmax search not always possible
  - Heuristic methods like beam search and reranking
How Does This Help?

- Independence assumptions can be relaxed
  - No need to estimate joint distribution $P(X, Y)$
  - Features over input $X$ come for free
- Prediction accuracy improves with rich features
  - Arbitrary combinations of input and output features
  - Fall back on heuristic inference for efficiency if needed
Problem Solved?

- Feature engineering
  - Feature combinations have to be hand-crafted
  - Feature selection requires trial-and-error experiments

- Sparse discrete features
  - Most features are binarized symbolic features (1-hot)
  - Feature vectors get extremely high-dimensional but sparse
  - Problematic for learning and efficient inference
Deep Learning in NLP (2014)

A Fast and Accurate Dependency Parser using Neural Networks

Danqi Chen
Computer Science Department
Stanford University
danqi@cs.stanford.edu

Christopher D. Manning
Computer Science Department
Stanford University
manning@stanford.edu

Abstract
Almost all current dependency parsers classify based on millions of sparse indicator features. Not only do these features generalize poorly, but the cost of feature computation restricts parsing speed significantly. In this work, we propose a novel way of learning a neural network classifier for use in a greedy, transition-based dependency parser. Because this classifier learns and uses just a small number of dense features, it can work very fast, while achieving an about 2% improvement in unlabeled and labeled attachment scores on both English and Chinese datasets. Concretely, our parser is able to parse more than 1000 sentences per second at 92.2% unlabeled attachment score on the English Penn Treebank.

of expertise and are usually incomplete. Third, the use of many feature templates cause a less studied problem: in modern dependency parsers, most of the runtime is consumed not by the core parsing algorithm but in the feature extraction step (He et al., 2013). For instance, Bohnet (2010) reports that his baseline parser spends 99% of its time doing feature extraction, despite that being done in standard efficient ways.

In this work, we address all of these problems by using dense features in place of the sparse indicator features. This is inspired by the recent success of distributed word representations in many NLP tasks, e.g., POS tagging (Collobert et al., 2011), machine translation (Devlin et al., 2014), and constituency parsing (Socher et al., 2013). Low-dimensional, dense word embeddings can effectively alleviate sparsity by sharing statistical strength between similar words, and can provide us a good starting point to construct features of
Chen and Manning’s Transition-Based Parser

- MaltParser with MLP instead of SVM (greedy, local)
- But 2 percentage points better LAS on PTB/CTB!?
Traditional Sparse Features

- Sparse – but lexical features and interaction features crucial
- Incomplete – unavoidable with hand-crafted feature templates
- Expensive – accounts for 95% of computing time

Indicator features

\[
\begin{align*}
    s_2.w &= \text{has} \land s_2.t = \text{VBZ} \\
    s_1.w &= \text{good} \land s_1.t = \text{JJ} \land b_1.w = \text{control} \\
    lc(s_2).t &= \text{PRP} \land s_2.t = \text{VBZ} \land s_1.t = \text{JJ} \\
    lc(s_2).w &= \text{He} \land lc(s_2).l = \text{nsubj} \land s_2.w = \text{has}
\end{align*}
\]

Stack
- ROOT has.VBZ good.JJ
  nsubj
Buffer
- control.NN

Binary, sparse dimension $= 10^6 \sim 10^7$

- **Motivation**
  - Model
  - Experiments
  - Analysis

- **Experiments**

- **Analysis**

- **Model**

- **Experiments**

- **Analysis**

- **Model**

- **Experiments**

- **Analysis**

- **Model**

- **Experiments**

- **Analysis**

- **Model**

- **Experiments**

- **Analysis**

- **Model**

- **Experiments**

- **Analysis**

- **Model**

- **Experiments**

- **Analysis**
Dense Features

- Sparse – dense features capture similarities (words, pos, dep)
- Incomplete – neural network learns interaction features
- Expensive – matrix multiplication with low dimensionality
PoS Embeddings

Machine Learning in NLP
Dep Embeddings
The Power of Embeddings

- One-Hot (discrete, sparse)
- Embedding (continuous, dense)

- Inherently much more expressive ($\mathcal{R} \times D$ vs. 1)
- Can capture similarities between items (sparsity)
- Can be pre-trained on large unlabeled corpora (OOV)
- Can be learned/tuned specifically for the parsing task
Recurrent Neural Networks

- Bi-LSTM encodes global context in word representations
- Character models capture morphology (and help sparsity)
Neural Network Techniques in Parsing

- Empirical results have improved substantially since 2014
- Neural network techniques yield more effective features:
  - Features are learned (not hand-crafted)
  - Features are continuous and dense (not discrete and sparse)
  - Features can be tuned to (multiple) specific tasks
  - Features can capture unbounded dependencies
  - Features can capture subword regularities
- Parsing architectures remain essentially the same
Strengths and Weaknesses

▶ Is (deep) machine learning always the solution?
Strengths and Weaknesses

▶ Is (deep) machine learning always the solution?

▶ On the one hand
  ▶ Learning from data is extremely powerful
  ▶ Normally the first choice for maximizing accuracy

▶ On the other hand
  ▶ Conditions for applying machine learning may not be ideal
  ▶ There may be additional factors to consider
The Unreasonable Effectiveness of Data?

- What kind of data is available?
  - Do we have labeled data?
  - How much data do we have?
  - Do we have data from the right domain/language?

- What to do if we don’t have adequate/sufficient data?
  - Collect and/or annotate (more) data
  - Apply cross-domain or cross-language learning
  - Consider a rule-based (or hybrid) method
F-score Isn’t All That Matters

- We may care more about minimum than average quality
F-score Isn’t All That Matters

- We may care more about minimum than average quality
- Users may want to have predictions explained
F-score Isn’t All That Matters

- We may care more about minimum than average quality
- Users may want to have predictions explained
- There may be ethical considerations with biased data
SOFTWARE SCANDALS
Prominent incidents that highlight the effect of algorithmic bias

December 2009 | Hewlett-Packard investigates instances of so-called "racist camera software" which had trouble recognizing dark-skinned people

March 2015 | A Carnegie Mellon University study determines that some personalized ads from sites such as Google and Facebook are gender-biased

July 2015 | A Google algorithm mistakenly captions photos of black people as "Gorillas"

March 2016 | Microsoft shuts down AI chatbot Tay after it starts using racist language

May 2016 | ProPublica investigation finds that a computer program used to track future criminals in the US is racially biased

September 2016 | Machine-learning algorithms used to judge an international beauty contest displays bias against dark-skinned contestants
F-score Isn’t All That Matters

- We may care more about minimum than average quality
- Users may want to have predictions explained
- There may be ethical considerations with biased data
- Companies often need to maintain legacy systems
Core NLP Technology Stack

- **Sentiment**
  - Machine Learning + Rules

- **Named Entities**
  - ML + Rules

- **Themes**
  - Rules

- **Topics**
  - ML + Rules

- **Summaries**
  - Rules

- **Intentions**
  - ML + Rules

- **Tokenization**
  - ML + Rules

- **PoS Tagging**
  - Machine Learning

- **Chunking**
  - Rules

- **Sentence Boundaries**
  - ML + Rules

- **Syntax Analysis**
  - ML + Rules

- **Sentence Chaining**
  - Rules

- **Concept Matrix**
  - Semantic Information

- **Syntax Matrix**
  - Syntax Information
Conclusion

- NLP today is overwhelmingly data-driven
- Deep learning is an evolution, not a revolution
- Machine learning is often the best solution
- But be open to pitfalls and alternative techniques