I design algorithms for intelligent processing of natural language texts—for example, to extract factual information into a structured database (e.g., extracting headquarters locations, CEOs, and phone numbers of companies from text into a database) or to predict real-world events from text (e.g., scientific trends, disease outbreaks). These applications require models of text that scale to large datasets. I advance machine learning (ML) methods for natural language processing (NLP), focusing on large-scale sparse models that leverage expert-informed domain knowledge. In my research, I seek to answer the following questions:

- Much like how humans draw conclusions, the best computational models are those that can leverage knowledge from various sources when performing inference. How can we better incorporate domain knowledge into machine learning models? For example, I improve NLP models using linguistically-motivated structured regularizers (§1) and incorporate temporal dynamics into ML models (§3).

- Scalability is an important issue in many NLP problems as NLP models become increasingly complex (hundreds of millions to billions of parameters). How do we efficiently learn such models on a large amount of data (billions of training examples)? I have developed learning algorithms for several very-large-scale real-world applications (§2).

- Machine learning is an integral part of text analysis. How do we formulate text understanding in a statistically sound manner? For example, in §4, I describe statistical methods for resolving and classifying named entity mentions in natural language texts.

I often leverage problem structure—both explicit structure in the input or output representation and implicit structure in the world that we want to capture in a model—to answer these questions. I have broad interests in many aspects of natural language processing and machine learning. In §5, I discuss future directions, including new forays into text visualization.

## 1 Structured Sparsity in NLP

Many learning problems can be formulated as a minimization of a sum of a loss function and a regularization function for a vector of parameters (feature coefficients). In NLP, much progress has been made on the loss function part (e.g., advances in structured prediction, latent variable models, and deep models). The regularization function plays two important roles: to prevent the models from capturing idiosyncrasies of the training data (overfitting) and to encode prior knowledge. However, past work rarely encodes non-trivial prior knowledge through a regularizer. In my research, I develop linguistically-motivated and cognitively-plausible regularization functions to advance NLP systems.

For example, in sentiment analysis, we know that not all sentences in a product review are relevant to the sentiment of the review. Many sentences merely describe the physical characteristics of the product. In Yogatama and Smith (2014b), I introduced a sentence regularizer that exploits this intuition to promote structurally sparse patterns at the sentence level. More generally, my method is applicable to many high-dimensional learning problems where only some parts of an observation are relevant to the prediction task. In Yogatama and Smith (2014a), I proposed three linguistically-motivated structured regularizers to complement the sentence regularizer. These can encode prior knowledge in the form of topical categories, hierarchical word clusters, and syntactic parse trees. I validated our structured regularizers by comparing them to standard unstructured regularizers that consider features in isolation, showing consistent benefits on a suite of datasets for various text prediction problems (topic classification, sentiment analysis, forecasting). This framework enables efficient incorporation of linguistic biases into conventional bag-of-words models without sacrificing convexity.

In another project (Yogatama et al., 2015), I introduced a regularizer that encourages hierarchical organization of word meanings when learning word representations. My work is the first that seeks
to regulate the relationships among dimensions of word embeddings. It is motivated by the coarse-to-fine organization of words’ meanings often found in the field of lexical semantics. For example, in the WordNet lexical database (Miller, 1995), words with close meanings are grouped together (e.g., professor and prof are synonyms), and fine-grained meaning groups are nested under coarse-grained ones (e.g., professor is a hyponym of academic). It also has a foundation in cognitive science, where hierarchical structures have been proposed as representations of semantic cognition (Collins and Quillian, 1969). In Figure 1, I show learned word representations for animal, horse, and elephant to illustrate our regularizer. In a quantitative evaluation, this approach outperformed state-of-the-art methods on various NLP benchmarks.

2 Large-scale Optimization

Modern ML models often do not straightforwardly scale to real-world problems. These problems can be large along several dimensions: the amount of training data, the number of parameters, or implicit problem structures (e.g., feature groups, hyperparameters). In my research, I bridge the gap between foundational work on optimization and its application (as a learning mechanism) to real-world problems by improving scalability of ML models across multiple dimensions above, often using NLP as an application area.

In Yogatama and Smith (2014b), I proposed an optimization method for sparse overlapping group lasso with thousands to millions of overlapping groups. This algorithm—based on the alternating directions method of multipliers (ADMM; Glowinski and Marroco (1975); Gabay and Mercier (1976)—is simple and efficient; it finds a reasonably good solution for most problems within tens of passes over the data. It is effective for a large number of overlapping groups because each group operation can be done in parallel. It also produces sparse solutions naturally, which is desirable for high-dimensional data such as text, unlike previous work with ADMM that requires heuristic thresholding (Qin and Goldfarb, 2012).

In another work (Yogatama et al., 2015), I introduced an efficient method for large-scale sparse coding with a structured penalty based on stochastic proximal methods. In sparse coding, the goal is to decompose an input matrix into two matrices: a dictionary matrix and a (sparse) code matrix. In modern settings, we are often interested in performing sparse coding for billions of training examples and hundreds of millions of parameters. Learning word representations from a corpus consisting of billions of tokens (e.g., our project in §1) exemplifies this setting. I introduced the first optimization method that scales to this problem size. This method can learn 200 million parameters reasonably well from a square input matrix of size 400,000 (160 billion entries) in a few hours on a regular 16-core machine.

When I interned at Google, I worked on hyperparameter optimization (i.e., model selection) when there are many hyperparameters to tune (e.g., 100). A hyperparameter is a parameter of a machine learning model that is fixed when learning the model (e.g., the number of training iterations, convergence tolerance). Having a large number of hyperparameters is increasingly common with the popularity of deep methods and ensemble methods. In many practical cases, an algorithm is required

Figure 1: Tree visualizations of word representations for animal (left), horse (center), elephant (right). Each box is a word embedding dimension. Red indicates negative values, blue indicates positive values (darker colors correspond to more extreme values). Dimensions that differ in sign mostly correspond to leaf nodes, validating our motivation that top level dimensions should focus more on general “characteristics” (for which they should be roughly similar for animal, horse, and elephant) and leaf nodes focus on word-specific “characteristics”.

(a) animal  
(b) horse  
(c) elephant

![Figure 1](image-url)
to find a good hyperparameter setting with few trials. For example, when providing a personalized machine learning service such as the Google Prediction API, it is unattractive in terms of user wait time and provider resource time to run many hyperparameter trials for each user. In Yogatama and Mann (2014), I proposed an efficient transfer learning method that accumulates information from previous datasets to find a good hyperparameter configuration faster. If $T$ is the number of models we select from for each dataset and $D$ is the number of datasets, the time complexity of our method is $O(T)$, while the best competing method is at least $O(DT^2)$. Experiments showed that our method scales to many more datasets without any loss in performance.

3 Temporal Analysis

Many datasets evolve rapidly based on events in the real world, but most NLP models can be categorized as “static” models—they are not aware of natural changes in the data over time. For example, trends within a scientific community are reflected in research articles published over time. Assuming a static model to predict responses to scientific articles becomes unrealistic. In general, more data is expected to improve predictions, but treating all past data equally risks distracting a model with irrelevant evidence. On the other hand, cautiously using only the most recent data risks overfitting to short-term trends and missing important time-insensitive effects.

In Yogatama et al. (2011), my collaborators and I proposed a Bayesian prior for time-dependent model parameters that leverages the intuition that the relationship between observable features and response variables should evolve smoothly over time. I further extended this prior to a general framework for learning model parameters that are sparse and adaptive to variation in how different parameters change over time (Yogatama et al., 2013). These priors can be used as structured regularizers in any generalized linear model.

In another project (Yogatama et al., 2014), I collaborated with a finance professor from the Tepper School of Business and others to design a probabilistic language model that captures temporal dynamics and conditions on arbitrary non-linguistic observations. For example, when building a language model for an economics application, we might want to use stock market data as exogenous evidence on which the model depends. When an important company’s price moves suddenly, the language model should reflect the language models for days when similar changes happened, since similar external stimuli are expected to lead to similar language use. For learning the model, I introduced an online variational expectation maximization algorithm to deal with non-stationarity in the data. This model achieved a 10% perplexity reduction on streams of economic news.

4 Information Extraction

An important aspect of text analysis is making inferences about entities mentioned in natural language documents (identifying, resolving, and classifying them). I designed an algorithm to construct a structured database of entities from an unstructured natural language text (Yogatama et al., 2012). For example, named entity mentions in a political blog dataset often refer to political figures (e.g., Mr. Obama, Barack Obama, Bill Clinton, Gov. Clinton, etc.). Given this kind of input, our model resolves Mr. Obama, Barack Obama to a single entity and Bill Clinton, Gov. Clinton to another entity. It also learns that {Mr., Gov.} and {Barack, Bill} are instances of similar entity attributes (titles and first names), which is important for creating structured databases of entities.

A newer algorithm (Yogatama et al., view) classifies named entity mentions into fine-grained labels (on the order of one hundred labels), taking into account the contexts they appear in. For example, in the sentence Hillary Clinton has written a new book in which she reveals she has never met President Barack Obama, “Hillary Clinton” should be labeled as an author since the sentence talks about her book and her role as an author of the book. In most other cases, “Hillary Clinton” is likely to be labeled a political figure. One difficulty in fine-grained entity classification is that we rarely have enough reliable training data. I proposed to overcome this problem by using a ranking loss and learning a joint representation of features and labels to allow information sharing for related labels. Our method achieved state-of-the-art results on entity-classification benchmarks.
5 Future Directions

Black box NLP  NLP researchers and practitioners spend a considerable amount of time comparing machine-learned models of text that differ in relatively uninteresting ways. For example, in categorizing texts, should the “bag of words” include bigrams, and is tf-idf weighting a good idea? These choices matter experimentally, often leading to big differences in performance, with little consistency across tasks and datasets in which combination of choices works best. In Yogatama and Smith (2015), I proposed an approach to optimizing text representations for text categorization problems. I see our work as a first step towards black box NLP systems that work with raw text and do not require manual tuning. Extensions to unsupervised learning, non-linear models, and structured prediction problems are exciting areas for future work.

Structured sparsity  I am also excited about the applications of sparse models—especially structured sparse models as a medium to encode prior knowledge. Nowadays, we often start with models with hundreds of millions to billions of parameters. Structured sparsity provides a way to completely discard some of these parameters in an informed and principled manner, resulting in smaller model size, which in turn leads to faster inference and easier interpretability. For example, mobile applications (e.g., Google Now, Siri, etc.) stand to benefit from smaller models since mobile phones typically have less storage and computing power than standard computers. Sparse models come with their own challenges. New varieties of sparse models often require a specialized optimization method, so I also intend to continue working on optimization methods for sparse models.

Big data  A related problem that is a major challenge in the era of big data is scalability. The amount of data continues growing and there is an increasing need to learn models faster. Many real-world applications involve billions of examples and parameters. Ongoing work on parameter server architectures (Li et al., 2014; Kumar et al., 2014) offers one potential solution, but I think the ability to do learning efficiently in a commodity computer architecture is also crucial for widespread application of personalized ML. I plan to continue my work on efficient learning methods for very-large-scale optimization problems.

Text visualization  A new area that I am interested in further pursuing is visualization of text data. The amount of text data is increasing, but we are not ready to face the problem of information overload. The ability to visualize contents of text data in a concise manner becomes very important. Text visualization goes hand in hand with NLP, since NLP provides a way to extract information to be visualized. For example, my work on sentence regularizer provides response-variable-specific summaries of text. I am interested in finding the best visualization methods such that readers can get information from an article in a short amount of time. From an NLP perspective, I think we can formulate text visualization as a text summarization problem, where the summary consists of visual units of analysis instead of textual units. For example, instead of compressing an article into a (smaller) set of sentences, we can compress it into a set of coloring schemes, visual icons, or even graphs. Predominant approaches to visualizing an article use words (http://www.wordle.net), phrases (van Ham et al., 2009), or topics (Blei et al., 2003) as the visual units. I think representation learning of variable-length pieces of texts (e.g., sentences, paragraphs) can help identify the best visual unit for an article and provide a principled way to incorporate spatial dimension into a visualization model. I see this area as an important intersection between NLP, ML, and visualization. Formulating the problem in this framework also opens the possibility to incorporate semantic cognition into a visualization model, building on my past and ongoing work on leveraging domain knowledge.

Summary  I think the future of natural language processing and machine learning will require specialized models that can leverage domain knowledge and harness the potential of big data. While general models that are applicable to various tasks are attractive due to their simplicity, they tend to underperform compared to task-specific models. What is currently missing is a general framework for automatically incorporating domain knowledge into very-large-scale ML and NLP models. I am excited to continue working on my long-term goal to build such a framework.
References


