CAREER: Flexible Learning for Natural Language Processing

Statistical learning is now a central part of natural language processing (NLP). Yet learning methods are seen as orthogonal to questions of linguistic representation, at best. At worst (and mistakenly in our view), statistical learning and linguistics are seen as incompatible. Bridging the gap requires learning that goes beyond **learning parameters**. This five-year project advances NLP by considering three questions:

1. Given recent advances in learning the parameters of linguistic models and in approximate inference, how can the process of feature design be automated? We seek **feature learning** methods that will find richer patterns for use in (supervised) statistical NLP, building on recent developments in supervised and unsupervised learning, and making more extensive use of available linguistic resources.
2. Given that (i) NLP tasks are often defined without recourse to real applications and therefore (ii) a specific annotated dataset is unlikely to fulfill the needs of more than a few NLP projects, we seek to extend learning frameworks to perform **automatic task refinement**, so that a linguistic analysis task may be simplified in some ways if it leads to more consistent, more precise, or faster performance in the context of a particular dataset, language, domain, and resource scenario.
3. Language is "about" the world. Can computational models of language take into account the **non-text context** (or metadata) in which our linguistic data are embedded? Building on recent success of our efforts in social text analysis and text-driven forecasting, we seek to exploit context to further refine models of linguistic structure while enabling advances in this new application area.

The theme of the proposal is *flexibility*: allowing a learner more degrees of freedom in feature choice, task definition, and sources of information.

Intellectual Merit

This project builds intellectual and formal connections between NLP and ML, between syntactic models and semantic models, between learning algorithms and representations (features), and between "deep" linguistic analysis and text-driven forecasting, a novel application.

Concretely, the project seeks to advance the state of the art on two central NLP "focus" tasks at very different stages of research. The first is multilingual dependency parsing, which has received considerable attention over the past few years. The second is frame-semantic parsing, which has received little attention and comes with relatively little data; it is a challenge that will have a large pay-off for many applications. These models are then used within the more experimental forecasting framework, where text is linked to measurable real-world states. Modeling text alongside context is expected to further advance performance on the focus tasks; we take first steps toward using forecasting to uncover linguistic knowledge.

Broader Impact

The research in this project is built on the foundations of statistical machine learning. We propose to develop structured regularizers, connect abstract notions of computational efficiency to modeling through features, extend classical notions of cost functions and output spaces, and seek to integrate feature learning with online algorithms. This view will enable the use of our contributions outside NLP, e.g., in computational biology and vision, where statistical models of structured data also play a key role.

Within NLP, this project contributes a new parser that directly models the most consistently learnable elements of syntactic struture. This parser will be designed for speed and applicability within real applications, where the high-level dependencies among key predicates and chunked base phrases are more important than complete structures conforming to a formal representation. This tool, as well as the more fine-grained syntactic and semantic analyzers developed in this project, will be made publicly available.

The project closely integrates research with education, including a new project based on text-driven forecasting within the PI's undergraduate NLP course, a new undergraduate course in machine learning. It supports involvement by the PI in outreach activities to high school students (through the North American Computational Linguistics Olympiad) and to a wider range of students at CMU by working with research collaborators to expose aspects of his research in non-CS classrooms. It will provide five years of graduate researcher support.

CAREER: Flexible Learning for Natural Language Processing

1 Introduction

Statistical learning is now a central part of natural language processing (NLP). Yet learning methods are seen as orthogonal to questions of linguistic representation, at best. At worst (and mistakenly in our view), statistical learning and linguistics are seen as incompatible. Bridging the gap requires learning that goes beyond **learning parameters**. This five-year project advances NLP by considering three questions:

1. Given recent advances in learning the parameters of linguistic models and in approximate inference, how can the process of feature design be automated? We seek **feature learning** methods that will find richer patterns for use in (supervised) statistical NLP, building on recent developments in supervised and unsupervised learning, and making more extensive use of available linguistic resources.
2. Given that (i) NLP tasks are often defined without recourse to real applications and therefore (ii) a specific annotated dataset is unlikely to fulfill the needs of more than a few NLP projects, we seek to extend learning frameworks to perform **automatic task refinement**, so that a linguistic analysis task may be simplified in some ways if it leads to more consistent, more precise, or faster performance in the context of a particular dataset, language, domain, and resource scenario.
3. Language is "about" the world. Can computational models of language take into account the **non-text context** (or metadata) in which our linguistic data are embedded? Building on recent success of our efforts in social text analysis and text-driven forecasting, we seek to exploit context to further refine models of linguistic structure while enabling advances in this new application area.

The theme of the proposal is *flexibility*: allowing a learner more degrees of freedom in feature choice, task definition, and sources of information. Because we build on strong machine learning (ML) foundations, the developments in this project will also be of use outside NLP, wherever models of structure are used (e.g., computational biology and computer vision).

2 Background and Notation

A major part of NLP involves the automation of linguistic analysis. We let $\mathbf{x}$ denote an input piece of text (sentence or document) and $\mathbf{y}$ denote a less ambiguous analysis of $\mathbf{x}$ (e.g., a syntactic or semantic parse of $\mathbf{x}$). $\mathcal{Y}_{\mathbf{x}}$ denotes the set of all possible analyses of $\mathbf{x}$; **linguistic structure prediction** is a general framework for building analyzers that select $\mathbf{y} \in \mathcal{Y}_{\mathbf{x}}$ given $\mathbf{x}$. Constructing a linguistic structure predictor, denoted here h, involves several considerations.

Representation. Most commonly, competing analyses in $\mathcal{Y}_{\mathbf{x}}$ are scored using a feature function $\mathbf{g}$ that maps $\langle \mathbf{x}, \mathbf{y} \rangle$ to a d-dimensional real-valued vector. The score of $\mathbf{y}$ (given $\mathbf{x}$) is given by $\mathbf{w}^\top \mathbf{g}(\mathbf{x}, \mathbf{y})$, where $\mathbf{w} \in \mathbb{R}^d$ are called weights and parameterize the model. This is known as a **linear model**, and it subsumes the vast majority of supervised NLP models.[1]

Decoding. Specifying the features that play a role in scoring competing analyses in $\mathcal{Y}_{\mathbf{x}}$ is not enough; we must also provide a procedure for choosing $h(\mathbf{x}) \in \mathcal{Y}_{\mathbf{x}}$. Linear models lead to a decoding (or "maximum inference") problem of the form

$$h(\mathbf{x}) = \operatorname*{argmax}_{\mathbf{y} \in \mathcal{Y}_{\mathbf{x}}} \mathbf{w}^\top \mathbf{g}(\mathbf{x}, \mathbf{y}) \tag{1}$$

Some decoding problems can be solved exactly and efficiently through dynamic programming or graph algorithms. Others are intractable and must be approximated using search or variational methods. This project does not focus on decoding, but does build heavily on recent decoding research by the PI and others.

Learning. In current NLP research, analyzers are most commonly constructed using a combination of expert knowledge (usually in designing the feature vector $\mathbf{g}$) and supervised learning from data (usually to select the weights $\mathbf{w}$). Generative approaches based on hidden Markov models and other probabilistic

[1] Many models are not described as linear models, but can be represented that way. Most models that are built from incomplete data can be understood as linear models with *hidden variables*. Models involving kernels can be understood as linear models where the features are implicitly represented by the kernel function.

grammars are a mainstay, though discriminative approaches generalizing logistic regression (conditional random fields; Lafferty et al., 2001), support vector machines (max-margin Markov networks; Taskar et al., 2004; Tsochantaridis et al., 2004), and the perceptron (structured perceptron; Collins, 2002) are widely used. All of these can be understood as solving empirical risk minimization problems of the form:

$$\mathbf{w}^* \leftarrow \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^d} R(\mathbf{w}) + \sum_{i=1}^{N} L(\mathbf{x}_i, \mathbf{y}_i, \mathbf{w}) \tag{2}$$

where R is a regularization term, N is the training dataset size, $\langle \mathbf{x}_i, \mathbf{y}_i \rangle$ is the ith training example, and L is a "loss" function that quantifies the model's performance on one example. Many losses and optimization algorithms have recently been developed for efficient optimization, including work by the PI (Gimpel et al., 2010; Martins et al., 2010a; Smith et al., 2007). Learning from *unannotated* data, either in combination with annotated data (semisupervised learning) or exclusively (unsupervised learning) can also be understood in the same framework (removing the $\mathbf{y}_i$).

Our monograph describes each of the above components in detail, with examples (Smith, forthcoming).

3 Focus NLP Tasks: Dependency Parsing and Frame-Semantic Parsing

We have claimed that linguistic structure prediction is a general framework applicable to many NLP problems (segmentation, sequence labeling, alignment, many kinds of parsing, entity resolution, etc.). Complex problems like summarization and translation can also be understood in this framework (Gimpel and Smith, 2009a; Martins and Smith, 2009).

In any particular research effort seeking central advances to NLP, there is a tension between focusing on the challenges unique to one important problem and seeking generality by applying techniques to *many* problems. For example, the PI's dissertation sought to make advances in unsupervised structure learning, focusing almost exclusively on dependency syntax (Smith, 2006). Having built a large research group (about ten at this writing), the PI has more recently moved in the direction of studying many problems at once, sharing ideas where appropriate. This style has led to state-of-the-art performance improvements on various tasks (e.g., multilingual dependency parsing, both supervised and unsupervised, frame-semantic parsing, named-entity recognition, answer selection for question answering, and paraphrase modeling) as well as generic techniques.

This project considers two "focus tasks": **syntactic dependency parsing** and **frame-semantic parsing**. These two tasks provide *test beds*, but our aim is to develop techniques that are broadly applicable across NLP, and we anticipate the use of these techniques on problems that arise in related, synergistic projects conducted within the PI's group (see §7). We have chosen these two tasks because (i) we have achieved state-of-the-art performance on both and are well-positioned to make further advances, (ii) they represent two very different points in the space of NLP problems, and (iii) they show promise for application to noisy text in under-studied domains (§4.3).

Dependency parsing. Since the 2006 and 2007 CoNLL shared tasks on multilingual dependency parsing, there has been a surge of research on this problem, and performance for many languages appears to be reaching a plateau. Yet for some languages—notably Arabic, Turkish, and Dutch—the best reported unlabeled dependency parsing performance remains below 85% attachment accuracy (performance is generally even lower for *labeled* dependency parsing). Our most recent results used a single method to achieve improvements over the best-published results on half of the standard datasets (Martins et al., 2010b), suggesting that there is still more to do. In particular, the best models are very large ($10^{7\sim8}$ features), making them expensive to use and inscrutable to humans. In this project we will explore ways to refocus this task that will result in more compact linguistic analyzers that are more focused to each particular language (by automatic methods) and that will be more useful for applications. Importantly, we note that research on this problem has consistently assumed that each parser will be learned in the exact same way (same feature representation insofar as the data provide the information, same decoding algorithm, same learner); we propose that giving the learner more flexibility in automatically managing features and trading off different kinds of errors, it will adapt better to the specifics of a dataset and language.

Frame-semantic parsing. Frame-semantic parsing, on the other hand, is a problem that has barely been studied. If successful, it offers great promise for semantic analysis, as the representation generalizes across

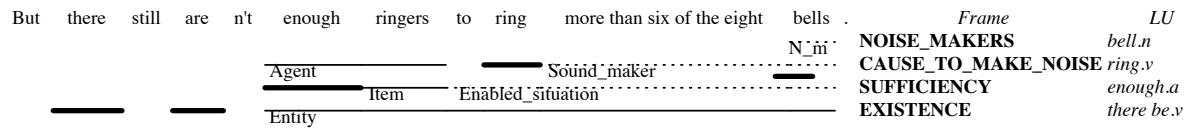

Figure 1: A sentence from PropBank and the SemEval 2007 training data, and a partial depiction of gold FrameNet annotations. Each frame is a row below the sentence (ordered for readability). Thick lines indicate targets that evoke frames; thin solid/dotted lines with labels indicate arguments. "N_m" under *bells* is short for the Noise_maker role of the NOISE_MAKERS frame. The last row indicates that *there...are* is a discontinuous target. Note that in PropBank, the verb *ring* is the only annotated predicate for this sentence, and it is not related to other predicates with similar meanings. Current state-of-the-art performance on this task is 62.8% precision and 41.9% recall (partial frame matching criterion, includes both frame and role disambiguation; Das et al., 2010).

related predicates and roles (unlike, e.g., PropBank, Palmer et al., 2005). It builds on FrameNet (Baker et al., 1998), a large-scale lexicon developed by linguists. The problem was introduced as a shared task at SemEval 2007 (Baker et al., 2007), a small dataset was released, and three teams competed. We recently published an approach (based on the structure prediction abstraction discussed in §2; Das et al., 2010) that substantially outperformed the best 2007 system (Johansson and Nugues, 2007); we then applied the system within a new SemEval task to find event participants *outside* the sentence where a predicate appears, achieving the best performance among competing systems, but with very modest scores (Chen et al., 2010).[2] The problem includes (i) disambiguating words (and phrases) to abstract *frames* (predicates) in the FrameNet lexicon and (ii) identifying the arguments that fill abstract *roles* for each frame (not unlike semantic role labeling). See Fig. 1 for an example structure. Current performance is very low; as currently defined, the task is almost certainly too difficult to be solved with the resources that have been developed. It therefore offers opportunities for innovation in learning and exploiting non-traditional resources beyond annotated data.

4 Technical Plan

This project explores three research directions, each of which involves algorithmic development and experimentation and aims to make learners for NLP more flexible. These directions are **feature learning** (§4.1), **automatic task refinement** (§4.2), and the exploitation of **non-text context** (§4.3).

4.1 Feature Learning

It is widely agreed that the most important part of any linguistic analyzer is the choice of features. Innovative features have led to many improvements in syntactic and semantic analyzer performance. In dependency parsing, for example, continued addition of new features has consistently led to better performance; most recently Koo and Collins (2010) showed performance gains using "third-order" features.

From a linguistic perspective, features encode a **lexicon**. Whereas traditional lexicons are lists of words or rules, feature functions can involve many heterogeneous bits of structure that "overlap" and interrelate. Lexicons have traditionally been built under specific theoretical assumptions (e.g., the lexicon for a context-free model of natural language syntax is a collection of production rules like "Sentence → Noun-Phrase Verb-Phrase"), but in featurized linear models, there is no theoretical impediment to including any element of structure at all. From a statistical modeling perspective, features define a **model family** that may or may not have sufficient expressive power to capture the phenomenon. From a learning theoretic perspective, features define the learner's hypothesis space.

The modularity of the approach described in §2 allows new features to be proposed and tested independently of other changes. There are, of course, practical difficulties. Features interact closely with decoding algorithms, and if features capture long-distance effects that link disparate parts of the structure $\mathbf{y}$, decoding

[2] This problem was also addressed in work receiving the ACL 2010 best paper award (Gerber and Chai, 2010), though in a different experimental setting.

can become prohibitively expensive. For example, features that link adjacent dependency arcs in a dependency tree make nonprojective dependency parsing NP-hard (McDonald and Satta, 2007), leading to very different decoding algorithms (Martins et al., 2009a). The other danger is **overfitting**; given a finite training dataset, we cannot expect a learned model to generalize well if there are too many free parameters to be set (here, the dimensionality d of $\mathbf{w}$ becomes large).

Recent research has led to the tentative conclusion that, with a reasonable approach to regularization,[3] adding more features to NLP models rarely harms performance. The standard approach is to specify "templates" for features (see Table 1 on page 11 for examples), then instantiate every feature that occurs at least T times in the training data. It has been reported that setting $T = 0$, i.e., including "unsupported" features not seen in the data but that can be inferred from alternative analyses of the training inputs $\mathbf{x}_i$, can lead to improvements in performance (Sha and Pereira, 2003); our recent results have corroborated that conclusion (Martins et al., 2010b). In short, current research primarily approaches features by manually specifying templates, instantiating those templates as features, then training weights $\mathbf{w}$ via Eq. 2.

In early work using feature-based models in NLP, automatic feature *selection* was seen as an important part of constructing a good model. Della Pietra et al. (1997), for example, described a greedy method for introducing new features to log-linear models by conjoining existing binary features incrementally. Despite some study in ML (see Guyon and Elisseeff, 2003 for an overview), feature selection has largely been ignored in NLP in the past decade, probably because of the conventional wisdom that more features never hurt, assuming we can develop efficient decoding techniques.

The status quo is that our models perform well but provide almost nothing in the way of human understanding about how they work or how language works. Analyzing our models usually comes down to browsing features with large-magnitude weights, ablation studies, and statistical analyses of system output, none of which help point the way toward deeper understanding of the task or the phenomenon, or even improved performance beyond the current model. Ideally, a model is easier to understand than the data from which it was constructed, and corresponds to a set of general, intelligible claims.[4]

In ML, one answer to this problem is to seek models that are **sparse**; i.e., where a small number of features have nonzero weights.[5] The simplest way to do this is to let $R(\mathbf{w}) = C\|\mathbf{w}\|_1 = C\sum_{j=1}^{d} |w_j|$, known as ℓ_1 or "lasso" regularization (Tibshirani, 1996). This has been explored in NLP, but results are not yet conclusive (Andrew and Gao, 2007; Goodman, 2004; Kazama and Tsujii, 2003). Sparsity offers the hope of smaller models that are more memory-efficient and human-browsable, but we believe there is more to be done in service of intelligibility.

It is time to revisit feature selection for structured NLP problems, for the following reasons:

- The relationship between decoding complexity and non-local features is now better understood through the framework of graphical models (Koller and Friedman, 2009). We can therefore begin to (i) use approximate inference when features make exact inference intractable and (ii) explicitly make use of runtime considerations when choosing features.
- Developments in regularization techniques beyond the ℓ_1 and ℓ_2 norms have opened up the possibility of treating more sophisticated feature selection within the central optimization problem solved by learning.
- Nonparametric techniques—those that allow the complexity of the model to grow based on the amount of available data—have matured and been integrated into structured modeling (Cohn et al., 2009; Nguyen et al., 2010).
- The development of new linguistic resources and new unsupervised techniques present opportunities for exploring a wide range of new features.

Note that learning features is not the same as learning structure in graphical models, an important problem but one that only focuses on statistical dependencies of random variables, not the parametric form underlying those dependencies (Koller and Friedman, 2009). An idea that relates the two is **constraints**, explored by Chang et al. (2007) among others, which can be understood as infinite-valued features, hard factors in graphical models, or restrictions on $\mathcal{Y}_{\mathbf{x}}$. Here we focus on feature learning.

[3] Most commonly, quadratic or "ℓ_2" regularization is used: $R(\mathbf{w}) = \frac{C}{2}\|\mathbf{w}\|_2^2$, with $C > 0$ tuned on development data.

[4] Along these lines, recent results in semisupervised learning have shown that human intuitions about specific, intelligible features can help a learner make better use of unannotated data (Druck et al., 2008).

[5] Setting a feature g_j's weight $w_j = 0$ is equivalent to excluding it from the model.

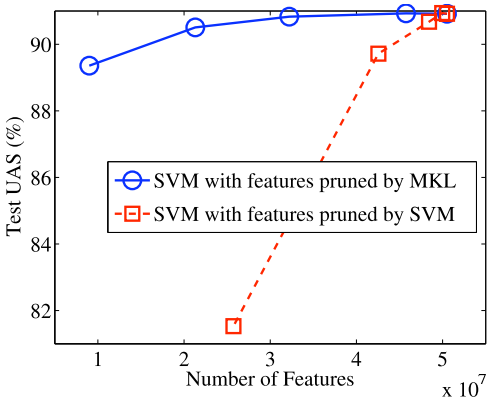

Figure 2: Structured SVMs for English dependency parsing trained on sets of feature templates of varying sizes, selected by SVM learning directly or MKL, in a first pass. In both cases, an SVM is used for learning after features are selected. UAS is the unlabeled attachment score, the fraction of non-punctuation words assigned the correct syntactic parent.

4.1.1 Structural Sparsity

Structural sparsity refers to the idea that the decisions about excluding different features are interdependent. For example, "group lasso" lets $\mathbf{w} = \langle \mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_k \rangle$, dividing weights into groups, and lets $R(\mathbf{w}) = \sum_{i=1}^{k} \|\mathbf{w}_i\|_2$, so that there is a benefit to setting any $\mathbf{w}_i = \mathbf{0}$, excluding a group of features collectively (Yuan and Lin, 2005). This has a close relationship to multiple kernel learning (MKL; Lanckriet et al., 2004), the details of which are out of scope but explored in detail in our recent work (Martins et al., 2010c).[6]

We seek to endow regularizers with knowledge about the structural relationships *among* features, so that a learner can make higher-level decisions about inclusion of feature-groups. A first step is to group features by the templates that they instantiate, so that whole templates can be excluded collectively. In a preliminary result, we found that using a group-sparse regularizer led to substantial reduction in the number of total features (by eliminating whole groups) with far better accuracy than models pruned using the SVM directly, after a second stage of SVM training on the selected features. See Figure 2.

We propose to explore regularizers that, for some groups, prefer within-group sparsity, allowing only a few features within a group to be used. This makes sense for groups of features that are non-local. Further, we seek to allow *overlapping* groups based on empirical attributes of features (like frequency in training data) and also formal attributes like subsumption by other features (i.e., more general features imply more specific ones; discouraging the use of specific features in the presence of a more general one may be advantageous for interpretability).[7]

The MKL framework provides a strong starting point for learning algorithms that are efficient and yet have convergence guarantees for norm-based regularizers $R(\mathbf{w})$. Such algorithms rely on alternately updating the weights based on its errors on a single example or small batch (as in the well-known perceptron algorithm and online subgradient methods; Collins, 2002; Ratliff et al., 2006) and updating the weights via **proximity operators** that respect the regularizer $R(\mathbf{w})$. Our approach generalizes FOBOS (Duchi and Singer, 2009) and truncated subgradient (Langford et al., 2009) and can be used to implement many structured regularizers.

What remains to be explored is how to use our understanding of the tasks to design those regularizers. In frame-semantic parsing in particular, considerable knowledge is encoded in the FrameNet lexicon that links frames (abstract concepts relating to events) and their roles; see Table 1 on page 11 and note the large degree of feature "overlap" and wide variation in specificity. Further, while the lexicon was designed by experts, it is neither complete nor perfect. A learner should be able to disregard some elements of its structure, and this can be accomplished through regularizers that encourage certain types of features to be ignored collectively if they are found to be unhelpful in most cases (group sparsity).

An alternative approach interprets the regularizer $R(\mathbf{w})$ as a log-(prior) probability of the choice of $\mathbf{w}$. This suggests the development of a probabilistic model over the feature lexicon, something attempted in

[6] A different take on sparsity is given by Graça et al. (2008), who consider sparsity in the *posterior* over hidden variables in unsupervised learning.

[7] Structured feature groups have been explored by Lee et al. (2007) for multitask learning (see also Argyriou et al., 2007) and Daumé (2007) for domain adaptation; other work by Chelba and Acero (2006) and Finkel and Manning (2009) explicitly used the regularizer in domain adaptation.

prior work in NLP (Eisner, 2002) and ML (Griffiths and Ghahramani, 2005). Optimizing among priors over $\mathbf{w}$ brings about learning challenges, since joint selection of $\mathbf{w}$ and R on the same data will lead to trivial results,[8] and not all regularizers can be treated under our MKL framework (e.g., it assumes $R(\mathbf{0}) = 0$). Cross-validation is an expensive technique to avoid this problem; we propose to develop, formally analyze, and empirically test online schemes that alternate between updates to the definition of R (based on held-out data) and to $\mathbf{w}$ and $\mathbf{g}$ (based on proper training data).

4.1.2 Feature Selection and Efficiency

We propose to develop regularizers $R(\mathbf{w})$ to penalize features that require greater computational expense or cause search errors in decoding. This idea has been explored in learning the structure of graphical models (Lowd and Domingos, 2008), but has yet to be seriously considered in structured modeling or NLP.

The central idea is to use the framework above (§4.1.1), but include in $R(\mathbf{w})$ an explicit model of the added expense of including a particular feature or group of features to the model. In dependency parsing, for example, we have explored the inclusion of sibling features (connecting a word and two of its children), grandparent features (connecting a word, its parent, and its parent's parent), and projectivity features, each of which make the required decoding algorithm considerably more expensive. In some situations, the benefit of adding such features may be outweighed by their computational demands. These runtime costs can be empirically estimated during the learning of the model (which calls decoding as a subroutine), allowing the regularizer to be adjusted as estimates of these runtime costs are obtained over many examples. As a result, we anticipate more control over the accuracy-runtime tradeoff.

Changing the regularizer during learning will endanger convergence guarantees, but we conjecture that estimates of feature runtime costs will converge quickly, so that the learner's objective function will be stable during most of the learning process.

4.1.3 Feature Induction in Online Learning

Online learning has come to be the dominant approach in structured modeling; this provides an excellent platform for mechanisms that propose new features on the fly. Introducing new features to a model means extending the vector function $\mathbf{g}$. In addition to conjunctive features considered by Della Pietra et al. (1997), *disjunctive* features can also be introduced.

Consider two binary features that match local structures, f_i and f_j.[9] The disjoined feature f is:

$$f(\boldsymbol{\pi}) = \begin{cases} 1 & \text{if } f_i(\boldsymbol{\pi}) = 1 \text{ or } f_j(\boldsymbol{\pi}) = 1 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where $\boldsymbol{\pi}$ is a local part of the overall structure (e.g., a clique in a graphical model). A concrete example is a hard clustering of words. If we have seven features involving lexical items *Sunday*, *Monday*, *Tuesday*, etc., for all of the days of the week—perhaps linking them to a temporal frame-semantic role or a syntactic position as a modifier to some verb—the disjunctive feature that combines all seven of the original features would effectively implement a "day of the week" word class feature, allowing a more general treatment by the model of all seven words together. This feature might then be combined, via traditional conjunction or disjunction, to create more features. The space of possible disjunctive features is exponentially large in the cardinality of the space of atomic features; any features can be disjoined, not just those that share a template.

We envision online learning algorithms, building on those discussed in §4.1.1, that explore new conjunctive and disjunctive features in tandem with weight learning, through well-formulated regularization functions $R(\mathbf{w})$. What features might be considered? Some possibilities follow.

Clusters. Recent results have shown that offline word clustering algorithms (e.g., Brown et al., 1992) can provide word classes that lead to performance improvements in parsing (Koo et al., 2008). Suzuki et al. (2009) further showed that estimating generative models on unannotated data can also provide features (generative model parameters) that enhance parsing performance. "Embeddings" are a related type of feature that has been found useful (Turian et al., 2010).

[8] This is equivalent to using the same data to estimate a language model and set smoothng parameters.

[9] Often elements of the vector function $\mathbf{g}$ are sums of local binary features, $g_i(\mathbf{x}, \mathbf{y}) = \sum_{\boldsymbol{\pi} \in \langle \mathbf{x}, \mathbf{y} \rangle} f_i(\boldsymbol{\pi})$.

Multi-word expressions. In frame-semantic parsing in particular, certain kinds of new features correspond to highly intuitive forms of linguistic knowledge. We envision feature induction that leads to conjunctive features corresponding to **multi-word expressions** (Sag et al., 2002) and **non-compositional compounds**, which are either provided from auxiliary methods or discovered on the fly during the learning procedure. Disjunctions of such features lead to a natural representation of **paraphrases**, which can be discovered and modeled using other supervised techniques (Das and Smith, 2009) and unsupervised techniques (Bannard and Callison-Burch, 2005; Barzilay and Lee, 2003, *inter alia*).

Predictions of simpler models. In a framework called **stacking**, simpler models' output can be used to provide new features for a learner. We explored this for dependency parsing (Martins et al., 2008), achieving state-of-the-art results and generalizing earlier ensemble approaches; we also gave an analysis of this technique as a way of *approximating* features for efficiency.

Expert-made resources. Another source of new features is linguistic resources such as lexicons and corpora annotated for purposes other than the task of interest. For example, we have found WordNet relations to be very useful in modeling question-answer structure (Wang et al., 2007), paraphrase structure (Das and Smith, 2009), and frame-semantic parsing (Das et al., 2010; see Table 1 on page 11). Features induced from expert-crafted resources like WordNet (Fellbaum, 1998), FrameNet (Baker et al., 1998), the Levin (1993) verb classes are a mainstay, but in the feature *induction* framework, they can be incrementally altered in service of the task. For example, the learner might decide to create a disjoined feature that effectively merges two WordNet synsets for better generalization in the context of a specific frame.

Decoding errors. In online learning, decoding (or related inference techniques) is a key subroutine, and updates to the model are made in response to errors made on each round. To date, only feature weight adjustments have been proposed; a novel extension is to design operators that introduce new features (conjunctive or disjunctive) based on errors. First we will consider model errors in scenarios where decoding is exact. Then, building on our recent development of algorithms that train weights to avoid search errors (Martins et al., 2009b), we will consider the introduction of features to correct *search errors* in approximate decoding scenarios, letting the learner correct for the failings of search.[10]

Consider the online update of our dual coordinate descent algorithm (Martins et al., 2010a) on iteration t, which generalizes MIRA (Crammer et al., 2006) for many more choices of L (N is the dataset size, $R(\mathbf{w}) = \frac{C}{2}\|\mathbf{w}\|_2^2$):

$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} - \underbrace{\min\left\{\frac{1}{NC}, \frac{L(\mathbf{x}_t, \mathbf{y}_t, \mathbf{w}_t)}{\|\nabla L(\mathbf{x}_t, \mathbf{y}_t, \mathbf{w}_t)\|^2}\right\}}_{\text{closed form "step size"}} \times \nabla L(\mathbf{x}_t, \mathbf{y}_t, \mathbf{w}_t) \tag{4}$$

In the hinge loss case, only features appearing in either the correct output $\mathbf{y}_t$ or the cost-augmented decoded output, $\hat{\mathbf{y}} = \operatorname{argmax}_{\mathbf{y}} \mathbf{w}^\top \mathbf{g}(\mathbf{x}_t, \mathbf{y}) + \operatorname{cost}(\mathbf{y}; \mathbf{x}_t, \mathbf{y}_t)$, will have their weights updated. Any feature g not yet included in the model can be understood as having weight $w = 0$. Therefore, on any iteration of online learning, new features appearing in $\mathbf{y}_t$, $\hat{\mathbf{y}}$, or both can be introduced using the update rule in Eq. 4. As noted, we have already explored this approach for the introduction of "unsupported" features (those that appear only in some $\hat{\mathbf{y}}$, not in any $\mathbf{y}_i$). The key research challenge is to define efficient operators for incrementally proposing new features of the kinds listed above. When taking structure into account, we propose that groups (e.g., feature templates) be introduced according to some schedule, each with a "trial period," past which the group weights are checked and some groups are removed.

4.2 Task Refinement through Learning

An important development in the past ten years has been the rise of **discriminative learning** for NLP problems, including the structured problems of interest here. The perceptron (Collins, 2002), conditional random fields (Lafferty et al., 2001), and various formulations of structured support vector machines (Taskar et al., 2004; Tsochantaridis et al., 2004) have had tremendous impact in NLP and have served to bring NLP and ML research closer together. Our recent contributions include a family of loss functions that properly includes the log-loss of CRFs and the structured hinge loss of structured SVMs (Gimpel and Smith, 2010), allowing the combination of probabilistic models with explicit task performance measures during learning, generic algorithms for "aggressive" online learning for many loss functions (Martins et al., 2010a), and asynchronous learning to exploit multiprocessor systems for faster convergence (Gimpel et al., 2010).

[10] A related idea is search-based learning (Daumé et al., 2009); there the focus is again on parameter learning.

Performance measures are often encoded as "cost" functions, where $\text{cost}(\mathbf{y}'; \mathbf{x}, \mathbf{y})$ is a nonnegative value signifying the badness of predicting output $\mathbf{y}'$ when the true output is $\mathbf{y}$ (given input $\mathbf{x}$). For example, the count of dependency attachment errors in $\mathbf{y}'$ is a cost function. We differentiate "cost" from "loss," which is a quantity the learning algorithm directly minimizes (L in Eq. 2), sometimes depending on the cost function (SVMs) and sometimes not (CRFs).

Some NLP problems have obvious and largely non-controversial cost functions (dependency attachment accuracy in parsing, for example), while others involve complex tradeoffs and more controversy (Bleu scores in machine translation; Papineni et al., 2002). Extensive research has been devoted to questions of automatic evaluation of NLP systems.

4.2.1 Learning the Cost Function

We propose to move beyond fixed loss functions $L(\mathbf{x}, \mathbf{y}, \mathbf{w})$ by allowing the learning algorithm to manipulate the cost measure. The notion of "cost" of labeling an input $\mathbf{x}_i$ with output $\mathbf{y}'$ rather than the correct answer $\mathbf{y}_i$ is central to discriminative learners like the structured SVM (Taskar et al., 2004):

$$L(\mathbf{x}_i, \mathbf{y}_i, \mathbf{w}) = -\mathbf{w}^\top \mathbf{g}(\mathbf{x}_i, \mathbf{y}_i) + \left(\max_{\mathbf{y} \in \mathcal{Y}_\mathbf{x}} \mathbf{w}^\top \mathbf{g}(\mathbf{x}_i, \mathbf{y}) + \text{cost}(\mathbf{y}'; \mathbf{x}_i, \mathbf{y}_i) \right) \tag{5}$$

Many cost functions (including those used to evaluate performance on the linguistic structure prediction tasks we consider) can be understood analogously to the feature functions used to score candidate outputs: $\text{cost}(\mathbf{y}'; \mathbf{x}, \mathbf{y}) = \mathbf{1}^\top \mathbf{c}(\mathbf{y}', \mathbf{x}, \mathbf{y})$. In other words, cost breaks down into a sum of local costs, usually for mismatches. In dependency parsing, we count the number of misattached words. Gimpel and Smith (2010) showed how precision and recall in named entity recognition could be traded off through the cost function.

Rather than weighting each mistake equally, we might parameterize the cost function so that different mistakes cost more or less. This is often done manually (e.g., ignoring dependency attachments for punctuation symbols). We propose to replace $\mathbf{1}$ in the expression above with a parameterized vector of weights $\mathbf{v}$, one for each different type of error,[11] and allow the learner to manipulate them. For the structured SVM case, this gives:

$$\min_{\mathbf{v} \geq \mathbf{0}, \mathbf{w}} T(\mathbf{v}) + R(\mathbf{w}) + \sum_{i=1}^{N} -\mathbf{w}^\top \mathbf{g}(\mathbf{x}_i, \mathbf{y}_i) + \left(\max_{\mathbf{y} \in \mathcal{Y}_\mathbf{x}} \mathbf{w}^\top \mathbf{g}(\mathbf{x}_i, \mathbf{y}) + \mathbf{v}^\top \mathbf{c}(\mathbf{y}, \mathbf{x}_i, \mathbf{y}_i) \right) \tag{6}$$

(It is easy to show that this is a convex problem that can be maximized using standard techniques like stochastic gradient descent or aggressive techniques like MIRA; Crammer et al., 2006; Martins et al., 2010a.) We discuss $T(\mathbf{v})$ below and in §4.2.2.

Of course, if the learner is provided total freedom to redefine the cost function, it may choose something trivial. Note, however, that setting $\mathbf{v} = \mathbf{0}$ in Eq. 6 results in a learning problem analogous to the structured perceptron (Collins, 2002), a reasonable method that often performs well. This means there is not an obvious way for a learner to "cheat" through the manipulation of the cost function through its parameters, $\mathbf{v}$. Nonetheless, we would like to encourage learning of models that perform well on some *consistent* parts of the task. For example, in frame-semantic parsing, we might prefer a model that achieves high accuracy for some frames or roles, while performing badly on others, to one that is consistently mediocre—though the learner gets to decide which frames and roles to focus on. A linear constraint, such as $\sum_j v_j = 1$, along with a $T(\mathbf{v})$ that encourages sparsity in $\mathbf{v}$, would encourage this kind of tradeoff.

This framework has important advantages. First, inspection of the learned cost weights $\mathbf{v}$ allows the researcher to see what parts of the cost function were attainable, after the fact. Large positive weights are a sign of confidence, while values $v_j = 0$ indicate where the model "gave up."[12] We envision very fine-grained notions of the cost function using rich features, providing a powerful new tool for error analysis. Second, a function $T(\mathbf{v})$ allows us to indicate preferences to the model by encouraging stronger emphasis on some elements of the cost function. This is analogous to regularization, only here we expect to encourage some

[11] E.g., in dependency parsing, noun and verb attachment errors could be counted as separate dimensions of $\mathbf{c}$.

[12] An alternative formulation of learner confidence, again emphasizing parameter learning, is **confidence-weighted learning** (Dredze et al., 2008), an orthogonal technique to what we propose.

v_j to be very large and positive. For example, in dependency parsing, correct attachment of a prepositional phrase might be noted as especially important by using a c-feature for matching PP attachments only and penalizing smaller values for its corresponding v-weight. Third, $T(\mathbf{v})$ might incorporate information from a downstream application, such as a summary of the most "dangerous" errors of this component affecting the system.

4.2.2 Learning the Output Space

The idea proposed above, to let learners "learn what they can learn" by manipulating the cost function, becomes much more interestng when we allow learners to manipulate the output space $\mathcal{Y}_{\mathbf{x}}$ as well. Because we consider structured problems, it is easy to open up the possibility of *partial* structures as outputs. This is an old idea; in parsing the idea of "partial parsing" or "chunking" was studied by Abney (1991) and shallow parsing is now a canonical NLP task. The new twist we propose is to allow the learner to empirically determine which kinds of structure should be sacrificed, for speed or accuracy.

To give an example, in Eisner and Smith (2005), we proposed "vine parsing," a variation on dependency parsing where long-distance attachments are simply disallowed. With hard bounds on the string distance between a word and its parent, dependency parsing runtime reduces asymptotically from $O(n^3)$ to $O(n)$ (where n is the sentence length; this reduction was observed empirically, too), and in Dreyer et al. (2006) we showed that not much overall accuracy needs to be sacrificed to take advantage of this kind of filter. The resulting structures are not connected dependency trees, but rather sequences of fragments. In some cases, where precision, consistency, and speed are required, this kind of approach may be attractive. Another line of work has sought to optimize more coarse-grained versions of an output space, in service of approximate decoding algorithms known as "coarse-to-fine" parsing (Charniak et al., 2006; Petrov et al., 2006).

We propose to (i) broaden $\mathcal{Y}_{\mathbf{x}}$ to include partial dependency structures and partial frame-semantic analyses, while (ii) enriching the cost function (discussed in §4.2.1) to encourage the learner not to "give up" on too many elements of structure, but if giving up is advantageous, to do so in a way that encourages *consistency*. Here $T(\mathbf{v})$ becomes extremely important; it can take on a structure that allows abandonment, for example, of some types of syntactic attachments or some types of frames or arguments, as a *group*.

We further envision a framework in which the learner is given a target value for a task-specific precision score (e.g., "95% precision must be obtained on unseen data"), and then the learner proceeds to tune $\mathbf{v}$ so that this target is obtained in a cross-validation setting, with the best possible task-specific recall.

We note that the ideas here are related to hidden variable learning by state splitting, as explored by Petrov (2009) and to work in finite-state output encodings explored by Loper (2008). Previous work has not, to our knowledge, considered an integrated view of cost functions, partial structures, and empirical risk minimization.

4.2.3 A New Parser for Applications

Many statistical parsers based on the Penn Treebank are available. We note, however, that few provide exactly what many NLP researchers need for downstream tasks and applications. A common phenomenon is that a researcher will run a parser, then transform its output by throwing away some elements of structure, or run two parsers and attempt to merge the outputs, or use one parser's postprocessing rules to augment the output of another parser (the Stanford dependency relations, described by de Marneffe et al., 2006, are a prime example; see also Johnson and Ural, 2010). The above framework provides an elegant way to train a parser that has some flexibility with regard to the training data, yet provides—in one integrated processing stage—consistent levels of structure that are more in line with what many researchers prefer.

We propose to design, implement, and evaluate a discriminative syntactic parser that models low-level chunks (like base noun phrases and prepositional phrases) with identified heads but without internal depenendencies, but uses labeled dependencies to connect major constituents in a consistent way that recovers key parts of the high-level structure (subject-verb and object-verb relations, PP attachment, relative clause structures, and so on). Rather than providing the exact desired outputs to the learner, we provide the (over-specified) complete structures (full treebank parses with heads), specify $\mathcal{Y}_{\mathbf{x}}$ to allow a wide range of partial parses for each example $\mathbf{x}$, and encode preferences about the cost function via $T(\mathbf{v})$. We plan to train this parser for all languages for which data are available.

An immediate application of the English version of our parser will be to our frame-semantic parser, which requires dependency parsing as preprocessing. Not all elements of the dependency parse are required for the frame-semantic parser, and sometimes dependency errors and conventions make reduce the ability of the frame-semantic parser to obtain high performance. Indeed, the argument identification module loses 19% of possible role-fillers because they do not correspond to subtrees in the parser output. Using the frame-semantic task, we can design cost functions that encourage the parser to find trees that (i) do not include elements of structure not needed for the task and (ii) lead to fewer errors in the semantics model.[13]

This idea is related to prior work on "loose" syntactic alignment (Burkett et al., 2010; Smith and Eisner, 2006; Wang et al., 2007), though it focuses on the analysis of single sentences directly, not an alignment or translation problem. It is also related to unsupervised approaches to segmentation (Cohn et al., 2009; Goldwater et al., 2006; Nguyen et al., 2010) that seek consistent representations of the data in service of likelihood. Instead of likelihood, here we have a notion of error, albeit a more flexible one than in traditional discriminative ML.

4.3 Non-Text Context

Recently, the PI, his students, and his colleagues have been exploring computational models that relate text to other kinds of data. A driving application of this idea is **forecasting**, or using text to make predictions about real-world states (for early work along these lines, see Ghose et al., 2007). We have demonstrated success on a wide range of forecasting problems: predicting return volatility of stocks (a measure of risk) from federally-mandated disclosures ("form 10K"; see Kogan et al., 2009), predicting opening weekend box office revenue of films from critical reviews (Joshi et al., 2010), and predicting commenting behavior in the blogosphere (Yano and Smith, 2010; Yano et al., 2009). Other studies have focused less on future predictions and more on correlations to other measures of real-world phenomena, e.g., using massive microblog streams to model public opinion (O'Connor et al., 2010) or using regional language variation to predict microbloggers' geographic locations (Eisenstein et al., 2010).

This line of work offers another motivating application for NLP, alongside the long-standing challenges of information extraction, machine translation, and summarization. Predicting real-world states has a big advantage for research: evaluation is often simple and need not rely on annotated data.

This research thread is somewhat distinct from the PI's work in structured NLP. This project, however, brings the two together. So far, research on forecasting has made use of shallow representations of text; bags of words, n-grams, and, in some cases, topics. Our general approach has been to tackle the problem through linear regression:

$$\min_{\mathbf{w}\in\mathbb{R}^d} R(\mathbf{w}) + \sum_{i=1}^{N} \left(\mathbf{w}^\top \mathbf{g}(\mathbf{x}_i) - m_i\right)^2 \tag{7}$$

where each $\langle \mathbf{x}_i, m_i \rangle$ is a text-input/state-output pair ($m_i \in \mathbb{R}$ is a scalar measurement like volatility or revenue), the $\mathbf{w}$ are coefficients, and $\mathbf{g}$ is a high-dimensional feature vector function encoding transformed frequencies of words, bigrams, and so on. We call this approach **text regression** (Kogan et al., 2009).

Text-driven forecasting problems are easy to find. Our research to date has led to three datasets (10-K reports, movie reviews, and political blogs, all publicly available to researchers), and conversations with social scientists and others have suggested many more. At this time we leave open the question of which forecasting problems will be of greatest interest in the next five years, or what kinds of new contextualized-text datasets will appear. We anticipate that, through continued collaboration with social scientists and exploration of new social media platforms, we will identify new forecasting scenarios in which to explore the ideas laid out here. The three datasets we have already constructed will serve as initial test beds, and we will add to this collection over time, continuing to make new datasets available to the research community. This project focuses on the basic research question of modeling the relationship of text to measurable world states.

[13] We note that the CoNLL 2008 and 2009 shared tasks focused on *joint* models for syntactic and semantic processing (Hajič et al., 2009; Surdeanu et al., 2008). This is an interesting direction, but so far results have been inconclusive. Joint inference is a challenging problem with many potential benefits (and one we have considered; Cohen and Smith, 2007; Martins and Smith, 2009), but for intelligibility and learnability, modular NLP components are more attractive, and that is what we pursue in this project. The developments here are also applicable, of course, to joint learning.

Features with both null and non-null variants: These features come in two flavors: if the argument is null, then one version fires; if it is overt (non-null), then another version fires.	
● some word in t has lemma λ	● some word in t has POS π
◐ some word in t has lemma λ, and the sentence uses PASSIVE voice	◐ some word in t has lemma λ, and the sentence uses ACTIVE voice
◐ the head of t has subcategorization sequence $\boldsymbol{\tau} = \langle \tau_1, \tau_2, \ldots \rangle$	◐ some syntactic dependent of the head of t has dependency type τ
● the head of t has c syntactic dependents	● bias feature (always fires)
Span content features: apply to overt argument candidates.	
○ POS tag π occurs for some word in s	
○ the head word of s has POS π	● $\lvert s \rvert$, the number of words in the candidate argument*
○ the first word of s has POS π, provided $\lvert s \rvert > 0$	○ the head word of s has syntactic dependency type τ
○ the last word of s has POS π, provided $\lvert s \rvert > 0$	● the syntactic dependency type τ_{s_1} of the first word with respect to its head
● the first word of s: w_{s_1}, and its POS tag π_{s_1}, provided that π_{s_1} is a closed-class POS†	● τ_{s_2}, provided that $\lvert s \rvert \geq 2$
● w_{s_2} and its closed-class POS tag π_{s_2}, provided that $\lvert s \rvert \geq 2$	● $\tau_{s_{\lvert s \rvert}}$, provided that $\lvert s \rvert \geq 3$
● the last word of s: $w_{s_{\lvert s \rvert}}$, and its closed-class POS tag $\pi_{s_{\lvert s \rvert}}$, provided that $\lvert s \rvert \geq 3$	○ the first word of s has lemma λ, provided $\lvert s \rvert > 0$
	○ the head word of s has lemma λ
	○ the last word of s has lemma λ, provided $\lvert s \rvert > 0$
◐ lemma λ is realized in some word in s	◐ lemma λ is realized in some word in s, the voice denoted in the span, s's position with respect to t (BEFORE, AFTER, or OVERLAPPING)
◐ lemma λ is realized in some word in s, the voice denoted in the span (ACTIVE or PASSIVE)	
Syntactic features: apply to overt argument candidates.	
○ dependency path: sequence of labeled, directed edges from the head word of s to the head word of t	○ length of the dependency path*
Span context POS features: for overt candidates, up to 6 of these features will be active.	
○ a word with POS π occurs up to 3 words before the first word of s	○ a word with POS π occurs up to 3 words after the last word of s
Ordering features: apply to overt argument candidates.	
● the position of s with respect to to the span of t: BEFORE, AFTER, or OVERLAPPING (i.e. there is at least one word shared by s and t)	○ target-argument crossing: there is at least one word shared by s and t, at least one word in s that is not in t, and at least one word in t that is not in s
○ linear word distance between the nearest word of s and the nearest word of t, provided s and t do not overlap*	○ linear word distance between the middle word of s and the middle word of t, provided s and t do not overlap*

Table 1: Feature templates for the argument identification module of our frame-semantic parser (Das et al., 2010), resulting in a feature set of 1,297,857 features—far more than are warranted by our current training dataset of 1,700 sentences. Every feature template has a version which does not take into account the role being filled (so as to incorporate overall biases). The ◐ symbol indicates that the feature template also has a variant which is conjoined with r, the name of the role being filled; and ● indicates that the feature template additionally has a variant which is conjoined with both r and f, the name of the frame. I.e., the ● symbol subsumes ◐, which in turn subsumes ○. The role name-only variants provide for smoothing over frames for common types of roles such as Time and Place; see Matsubayashi et al. (2009) for a detailed analysis of the effects of using role features at varying levels of granularity. *Quantized into groups: $(-\infty, -20]$, $[-19, -10]$, $[-9, -5]$, -4, -3, -2, -1, 0, 1, 2, 3, 4, $[5, 9]$, $[10, 19]$, $[20, \infty)$. †We treat as a closed-class POS tag any Penn Treebank tag except for CD which does not start with V, N, A, or R.

prediction (m)	text ($\mathbf{x}$)	error	baseline	only text	combined
return volatility for a firm (Kogan et al., 2009)	10-K report	mean squared error	0.1655	0.1667	**0.1538**
opening weekend per-screen revenue of a film (Joshi et al., 2010)	critical reviews	mean absolute error	\$6540	**\$6010**	**\$5998**
geographic location of a microblogger (Eisenstein et al., 2010)	Twitter feed	median error	1015 km	**501 km**	not applicable

Table 2: Selected recent results using text regression. "Baseline" does not use text (respectively, it uses historical volatility information, structured movie metadata, or a naïve central location).

4.3.1 Enriched Features for Text Regression

Most of what we said in §4.1 about learning features is applicable in the text regression case. We expect that phrasal fragments, multi-word expressions, idioms, proper names, syntactic fragments, and semantic frames and role mentions all might serve as useful abstractions from word features. n-grams are a crude approximation and have consistently been useful, but n could always be larger. For instance, *'t get me* and *get me wrong* are strong positive revenue features in box office prediction (Joshi et al., 2010), indicating that the phrase *don't get me wrong* is an important non-compositional discourse marker. Another case in 10-K reports is *a going concern*, deduced to be a strong sign of volatility (Kogan et al., 2009), though there are many alternative phrasings of this idea. Recently, Nakagawa et al. (2010) gave evidence that linguistic structure is important for sentiment analysis, a related high-level text understanding problem.

The obstacles to the use of such a broad set of features include:

- The text is often out-of-domain for a syntactic or semantic parser, and parser performance may be quite bad. It is unclear how this will affect prediction performance; a crude sentiment analyzer was sufficient to obtain strong correlation with other public opinion measures (O'Connor et al., 2010).
- Beyond mere domain, social text is often noisy, full of misspellings and nonstandard punctuation that will confound traditional linguistic analysis models.
- The space of features may be prohibitively large. We have found that very few regression packages are up to the challenge when d and N are both large (in the millions and thousands, respectively). Hence online feature selection techniques (§4.1.3) will be necessary.

We propose to develop methods that incrementally add syntactic and semantic features during online learning of text regression models. These techniques will be similar to those proposed in §4.1, but here the output space is $\mathbb{R}$ (measurements of world states). Linguistic structure predicted by syntactic and semantic analyzers ("$\mathbf{y}$" in our notation) becomes part of the input to the model. Hence the dimensions of the problem are very different. Structured regularizers (§4.1.1) are expected to play a central role in the development of efficient methods for coping with very large numbers of features (d); automatic feature discovery (§4.1.3) will serve to identify specific phrases and semantic frame invocations that are particularly informative about the world state to be predicted (m).

In large-data cases, we conjecture that noisy data and parsing errors (e.g., as in social media like blogs, blog comments, and microblogs) will not have much effect on performance, since precision is far more important than recall. (Note that models that perform only parts of the task they can achieve reliable performance on may be especially useful here; see §4.2.2.) In more focused scenarios involving a single document as input (like the 10-K scenario), noise is less of a problem, but the language may be more complex and domain issues more pressing. By considering a range of problems, we expect to gain a better understanding of appropriate linguistic structure for reasoning about future world states in general, while permitting specialization to achieve strong performance in any given scenario.

4.3.2 Closing the Loop

We propose that learning models that link linguistic structure to other streams of data (financial, political public opinion, and behavioral) can shed light on language itself. Indeed, we draw inspiration from research on NLP in the **biological domain**, where rich structured biological ontologies and considerable focused effort have led to advances focused on the unique problems of language in that domain (Cohen, 2007). Relatedly, models of text and risk (Kogan et al., 2009) might be useful, for instance, in "risk mining" (Leidner and Schilder, 2010). At a more basic level, Clarke et al. (2010) recently showed how external signals could improve semantic parsing into logical forms.

When we consider focused forecasting problems, we are often faced with novel genres, styles, and topics, so that learning models that link text features to extrinsic world states has the potential to broaden the language our linguistic analyzers can handle, especially if we consider a wide range of forecasting problems. In other words, we hope for models of the mapping from text $\mathbf{x}$ to world state m will also help reveal the elusive linguistic structure $\mathbf{y}$. By analogy, **contrastive estimation** (Smith and Eisner, 2005) used notions of well- and ill-formedness of sentences to reveal hidden syntactic structure in unsupervised grammar induction, with very strong results. More recently, Chang et al. (2010) have used high-level tasks to reveal hidden linguistic and alignment structure through what they call "indirect supervision."

While the Company is widely recognized as a leading innovator in the personal computer and consumer electronics markets as well as a leader in the emerging market for distribution of digital content, these markets are highly competitive and subject to aggressive pricing.

Figure 3: An excerpt from Apple Inc.'s Form 10-K filed in November 2008. The word *distribution* is in FrameNet as a lexical unit for the DISPERSAL frame; here its arguments are *the Company* (Apple) as Agent and *digital content* as Individuals. At present, FrameNet's lexicon does not provide an economic interpretation for *distribution* by linking this frame to any kind of economic activity (e.g., COMMERCE_SCENARIO). Note also that *personal computer*, *emerging market*, and *digital content* may be important multiword units in this domain.

We propose to use the features discovered and selected for forecasting models to augment syntactic and frame-semantic parsers. We believe that features (e.g., word and phrase classes, substructures of frame-semantic parses) that are discovered for the purpose of forecasting may lead to more robust models for certain frames and roles relevant to the forecasting domain. In longer term research (beyond this project), we envision automated discovery through forecasting tasks to aid humans in the further development of resources like FrameNet (see Figure 3).[14]

The starting point is to use automated methods to build feature-rich models for a forecasting problem (or multiple problems), then propose those features within a linguistic structure prediction model. It is an open question whether to use the feature *weights* from the forecasting model. One way to do so would be to group these forecasting-derived features by their weights, and inject them as grouped features (see §4.1.1) so that their effects would naturally move together in the linguistic prediction model. One view of this direction is as a technique for domain adaptation from a domain in which annotated data are available (in the frame-semantic case, fiction, light non-fiction, and government reports) to a new one (e.g., financial reports or blogs). Another view is that extracting features useful in forecasting is a way to seed a domain-specific lexicon, to which further structure can be added manually by experts or automatically by bootstrapping learners. This project takes first steps toward exploiting forecasting tasks as a source of linguistic knowledge.

4.4 Summary

This project seeks to increase the flexibility of learning methods in NLP. By automating the management of features (§4.1) and the cost structure of NLP tasks (§4.2), we will increase the flexibility of computational language learners faced with new domains, languages, and application settings. Further, by extending our models to link language data with other data, we anticipate advances in both domain-specific linguistic representations and a novel application area (§4.3).

5 Education and Outreach Plan

In his first few years as a faculty member, the PI has poured energy into educational activities at the graduate and undergraduate levels that are integrated with his research. He introduced two new research seminars and a new annual graduate course on statistical NLP. This course was consolidated into an invited tutorial at ICML 2009 and the material will be published as a monograph in the near future (Smith, forthcoming).

Teaching undergraduates is fundamentally different from teaching graduate students, in course content, depth-breadth tradeoffs, assignment and exam structure, and in lecture style. The PI has designed a new undergraduate-only, introductory NLP course, and has taught it annually (three times). The course includes a competitive project on question generation and answering (Smith et al., 2008; twice the project included teams from HHHH's analogous course at the University of Pittsburgh); that project led to a dissertation topic on automatic factual question generation for the PI's advisee (and the course's inaugural teaching assistant) MMMMM. The PI has advised 11 undergraduates on research projects, mostly drawn from the NLP course and often supported by NSF REU supplements. This has led to three senior theses and two students as co-authors on top-tier publications (LLLLLLL on Kogan et al., 2009, and CCCCCC on Das et al., 2010). Two NLP course alumni joined the PI's group as programmers after graduation.

[14] The PI has agreed to serve in an advisory capacity to a forthcoming proposed project, led by PPPPP , that will use learning to support expansion and integration of lexical semantics resources.

The PI plans continued annual offering of the undergraduate, project-based NLP course, with the introduction of a new competitive course project based on a forecasting problem. In addition, he will continue to invest in undergraduate research projects. Importantly, experience has shown that engaging students *early*—well before the senior year—leads to greater chances of real research contributions by the student. A semester or year is not enough, and students cannot be expected to contribute meaningfully to research or learn research skills without spending time in a supporting role first. This means active recruiting and long-term engagement. Coordinating with colleagues in the Language Technologies Institute, the PI hosts a lunch each semester with undergraduates, to give a short talk about language technologies and demonstrations of current LTI projects. In April 2010, the PI gave a tutorial on NLP to undergraduates at CMU's Qatar campus in Doha. **These activities will continue.**

Like every academic in CS today, the PI is concerned by our failures to attract to CS to more students, more *diverse* students, and to do so earlier, during the K–12 years. The PI does not wish to propose new initiatives that are outside his domain of expertise, but rather seeks to leverage the inherent accessibility of human language, which everyone uses, and the obvious usefulness of language technologies to create new entry points into CS for a wider range of students. In this project, the PI proposes the following:

1. North American Computational Linguistics Olympiad. NACLO is an effort sponsored by the NSF to engage high school students with the field. It is led by LLL , a colleague in the PI's department. The PI has had involvement in NACLO in past years (outreach, training sessions, and grading) and is well-positioned to increase his role. In addition, the PI will perform a statistical analysis of past student performance and demographics, to help determine ways to select problems that are more accessible to a more diverse set of students. Prof. L has voiced strong support for the PI's involvement.

2. ML for undergraduates. The PI plans to partner with colleagues in the Machine Learning Department (OOO and PPP) to develop a new project-based course sequence in ML specifically for undergraduates, building on the success and lessons of the NLP course. In our experience, undergraduates see ML as an exciting and important area of CS, but one with a high barrier to entry.

3. Outreach outside CS. The PI seeks novel ways to teach elements of computer science to non-majors; three likely routes follow. (i) The PI has helped to develop instructional materials for a course on rhetoric taught by MMM (CMU's English Department). Specifically, the PI has used NLP models for content selec-tion and data preparation to support a pedagogical project on political discourse. A lecture by the PI in that course is planned, surveying NLP and explaining the tools used to construct the dataset. (ii) Building on research collaboration with NNN (CMU's Tepper School of Business), we envision an interdis-ciplinary course that ties together models of economic behavior and of language, through real-world data.(iii) The PI's spouse is TTTTTTT (XXXX's Computatational Biology and Biological Sciences Departments); she has unique insight into the role of computing in basic biological research. Building on many years of conversations with her about how computational thinking from CS and the day-to-day empiricism of the wet lab are mutually reinforcing ideas, educational projects are expected to develop.

6 Results of Prior NSF-Sponsored Research

Smith has received two short-term NSF awards: "Parsing models and algorithms for morphologically rich languages" (IIS-0713265, 2007–9, $100K plus REU) and "SGER: Scaling up unsupervised grammar induction" (IIS-0836431, 2008–9, $197K plus REU). The second project led to the release of software tools for unsupervised and supervised dependency parsing, frame-semantic parsing, and the aforementioned blog dataset (see `http://www.ark.cs.cmu.edu`). It also supported Ph.D. student DDD's summer visit to Princeton University to start a collaboration between the PI and XXX. Smith is the co-PI on "INCA: an integrated cluster computing architecture for machine translation" with XXX (IIS-0844507, 2009–11, $450K plus REU), which focuses on the use of multiprocessor systems in machine translation com-ponents. A new open-source machine translation decoder has been released under this project. Smith is the PI on "Probabilistic models for structure discovery in text" (IIS-0915187, 2009–12, $450K plus REU). This project seeks to bring together linguistic knowledge with unsupervised learning, extend the reach of unsu-pervised structure discovery in NLP, and automate some of the more tedious aspects of unsupervised NLP. There are clear synergies, though the emphasis here is on flexible *supervised* learning.

Smith's Ph.D. advisees AAA and BBB have been primarily supported by the above projects, and at various times so have CCC and DDD NSF projects have also enabled collab-

orations with XXX, NNN, YYY, and their students. Research outcomes from these projects have been reported in high-profile NLP and ML forums (Cohen and Smith, 2009a,b, 2010; Cohen et al., 2008, 2009, 2010; Das et al., 2010; Gimpel and Smith, 2008, 2009a,b, 2010; Gimpel et al., 2010; Martins et al., 2008, 2009a,b; Nguyen et al., 2010; Smith et al., 2008; Venugopal et al., 2008, 2009; Yano et al., 2009). Each project was supplemented by an REU; over the past four years, the PI has involved six students in NSF research projects through REUs, and several more through academic independent study.

7 Synergistic Projects

The PI holds weekly group meetings, encouraging collaboration among students across projects and sharing of ideas across problems. The PI is currently involved in several non-NSF sponsored research projects that are synergistic with this project. (i) One is a collaboration with ZZZ (CMU–Qatar), sponsored by the Qatar National Research Foundation, explores cross-lingual and semisupervised learning to improve Arabic NLP using raw data and English resources. Parsing improvements for English and Arabic resulting from this project will support that effort, and we expect some shared code. (ii) Pending negotiations, a collaboration with SSS (CMU's Tepper School of Business) on text-driven forecasting for scientific literature, sponsored by IARPA, will begin soon. That effort is expected to provide new datasets and prob-lems to be considered in the contextualized NLP part of this project. (iii) The PI co-advises a student, EEE, in the CMU-Portugal dual Ph.D. program. Regular visits to Lisbon by the PI are part of this pro-gram, a result is ongoing ML collaboration with QQQ and RRR at IST. (iv) The PI plays a supporting role in the Google-sponsored "Worldly Knowledge" project led by TTT (CMU). That project focuses on automatic knowledge base construction from the Web, particularly identifying social, temporal, and geographical context for extracted facts; it is highly synergistic with §4.3.

8 Timeline

This project involves the PI and, at any given time, one graduate student; different students will likely be involved in the project at different times. Teaching plans are subject to approval by the PI's departments. The timeline below describes proposed activities during each year.

Annual activities. Teach undergraduate NLP and graduate NLP seminar (spring); in alternating fall semesters, teach graduate course on structured prediction or undergraduate ML. Support NACLO with outreach, training, and/or grading (§5, #1).

2011. Research focus on structural sparsity (§4.1.1), including sparsity driven by computational efficiency (§4.1.2), with evaluations on dependency parsing and frame-semantic parsing. Gather NACLO data and perform demographic analysis, providing recommendations to the organizers (§5, #1).

2012. Research focus on online feature induction (§4.1.3) for dependency parsing and frame-semantic parsing, as well as forecasting (§4.3.1). Summer: release of new versions of code for dependency and frame-semantic parsing to the research community. Fall: introduce new undergraduate ML course (§5, #2).

2013. Research focus on learning cost functions (§4.2.1) and output spaces (§4.2.2) for multilingual dependency parsing. Development of a new parser that focuses on the most useful elements of syntactic structure (§4.2.3). Continued use of new techniques in text regression on new problems as they arise. Spring: Introduce new forecasting project in undergraduate NLP course. Summer: release of new versions of code for dependency and frame-semantic parsing to the research community, as well as the new parser.

2014. Research focus on frame-semantic parsing, adapting learning techniques for the cost function (§4.2.1) and output space (§4.2.2) for this particular task, and seeking a more general treatment applicable to a wider range of problems. Continued use of new techniques in text regression on new problems as they arise. Summer: release of new versions of code for dependency and frame-semantic parsing to the research community, including the new parser. Fall (or sooner): new course, mini-course, or other programmatic interdisciplinary effort to expose non-CS students to CS (§5, #3).

2015. Extraction of features from forecasters, and use within syntactic and frame-semantic models directly (§4.3.2). Fall: final release of code for dependency and frame-semantic parsing to the research community.