University of Pennsylvania
ScholarlyCommons

Departmental Papers (CIS) Department of Computer & Information Science

5-1-2010

Stuctured Predictions Cascades

David Weiss
University of Pennsylvania

Ben Taskar
University of Pennsylvania, taskar@cis.upenn.edu

Structured Prediction Cascades, D. Weiss and B. Taskar. International Conference on Artificial Intelligence and Statistics (AISTATS), Sardinia, Italy, May 2010.
Copyright is held by the authors.

This paper is posted at ScholarlyCommons. http://repository.upenn.edu/cis_papers/546
For more information, please contact repository@pobox.upenn.edu.

Structured Prediction Cascades

David Weiss and **Ben Taskar**
University of Pennsylvania

Abstract

Structured prediction tasks pose a fundamental trade-off between the need for model complexity to increase predictive power and the limited computational resources for inference in the exponentially-sized output spaces such models require. We formulate and develop *structured prediction cascades*: a sequence of increasingly complex models that progressively filter the space of possible outputs. We represent an exponentially large set of filtered outputs using max marginals and propose a novel convex loss function that balances filtering error with filtering efficiency. We provide generalization bounds for these loss functions and evaluate our approach on handwriting recognition and part-of-speech tagging. We find that the learned cascades are capable of reducing the complexity of inference by up to five orders of magnitude, enabling the use of models which incorporate higher order features and yield higher accuracy.

1 Introduction

The trade-off between approximation and estimation error is fundamental in learning complex prediction models. In structured prediction tasks, such as part-of-speech tagging, machine translation and gene prediction, the factor of computation time also plays an important role as models with increasing complexity of inference are considered. For example, a first order conditional random field (CRF) (Lafferty et al., 2001) is fast to evaluate but may not be an accurate model for phoneme recognition, while a fifth order model is more accurate, but prohibitively expensive for both learning and prediction. (Model complexity can also lead to overfitting problems due the sparseness of the training data. We do not specifically address this problem in our paper other than judiciously regularizing model parameters.)

In practice, model complexity is limited by computational constraints at prediction time, either explicitly by the user or implicitly because of the limits of available computation power. We therefore need to balance expected error with inference time. A common solution is to use heuristic pruning techniques or approximate search methods in order to make higher order models feasible. While it is natural and commonplace to prune graphical model state space, the problem of explicitly learning to control the error/computation tradeoff has not been addressed. In this paper, we formulate the problem of learning a cascade of models of increasing complexity that progressively filter a given structured output space, minimizing overall error and computational effort at prediction time according to a desired tradeoff. The contributions of this paper are:

- A novel convex loss function specifically geared for learning to filter accurately and effectively.
- A simple online algorithm for minimizing this loss using standard inference methods.
- Theoretical analysis of generalization of the cascade (in terms of both accuracy and efficiency).
- Evaluation on two large-scale applications: handwriting recognition and part-of-speech tagging.

2 Related Work

Heuristic methods for pruning the search space of outputs have been exploited in many natural language processing and computer vision tasks. For part-of-speech tagging, perhaps the simplest method is to limit the possible tags for each word to those only seen as its labels in the training data. For example, the MXPOST tagger (Ratnaparkhi, 1996) and many others use this technique. In our experiments, we compare to this simple trick and show that our method is much more accurate and effective in reducing the output space. In parsing, the several works (Charniak, 2000; Carreras et al., 2008; Petrov, 2009) use a "coarse-to-fine" idea

Appearing in Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS) 2010, Chia Laguna Resort, Sardinia, Italy. Volume 6 of JMLR: W&CP 6. Copyright 2010 by the authors.

closely related to ours: the marginals of a simple context free grammar or dependency model are used to prune the parse chart for a more complex grammar. We also compare to this idea in our experiments. The key difference with our work is that we explicitly learn a sequence of models tuned specifically to filter the space accurately and effectively. Unlike the work of Petrov (2009), however, we do not learn the structure of the hierarchy of models but assume it is given by the designer.

It is important to distinguish the approach proposed here, in which we use *exact* inference in a *reduced* output space, with other *approximate* inference techniques that operate in the full output space (e.g., Druck et al. (2007), Pal et al. (2006)). Because our approach is orthogonal to such approximate inference techniques, it is likely that the structured pruning cascades we propose could be combined with existing methods to perform approximate inference in a reduced output space.

Our inspiration comes partly from the cascade classifier model of Viola and Jones (2002), widely used for real-time detection of faces in images. In their work, a window is scanned over an image in search of faces and a cascade of very simple binary classifiers is trained to weed out easy and frequent negative examples early on. In the same spirit, we propose to learn a cascade of structured models of increasing order that weed out easy incorrect assignments early on.

3 Structured Prediction Cascades

Given an input space $\mathcal{X}$, output space $\mathcal{Y}$, and a training set $S = \{\langle x^1, y^1\rangle, \ldots, \langle x^n, y^n\rangle\}$ of n independent and identically-distributed (i.i.d.) random samples from a joint distribution $D(X, Y)$, the standard supervised learning task is to learn a hypothesis $h : \mathcal{X} \mapsto \mathcal{Y}$ that minimizes the expected loss $\mathbb{E}_D\left[\mathcal{L}\left(h(x), y\right)\right]$ for some non-negative loss function $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$.

We consider the case of structured classification where Y is a ℓ-vector of variables and $\mathcal{Y} = \mathcal{Y}_1 \times \cdots \times \mathcal{Y}_\ell$. In many settings, the number of random variables Y differs depending on input X (for example, length of the sentence in part of speech tagging), but for simplicity of notation, we assume a fixed number ℓ here. We denote the components of y as $y = \{y_1, \ldots, y_\ell\}$, where $y_i \in \{1, \ldots, K\}$. The linear hypothesis class we consider is of the form:

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

where $\mathbf{w} \in \mathbb{R}^p$ is a vector of parameters and $\mathbf{f} : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}^p$ is a function mapping (x, y) pairs to a set of p real-valued features. We further assume that $\mathbf{f}$ decomposes over a set of cliques $\mathcal{C} \subseteq \mathbb{P}\{X, Y\}$ (where $\mathbb{P}$ is a powerset):

$$\mathbf{w}^\top \mathbf{f}(x, y) = \sum_{c \in \mathcal{C}} \mathbf{w}^\top \mathbf{f}_c(x, y_c). \tag{2}$$

Above, y_c is an assignment to the subset of Y variables in the clique c and we will use $\mathcal{Y}_c$ to refer to the set all assignments to the clique. By considering different cliques over X and Y, $\mathbf{f}$ can represent arbitrary interactions between the components of x and y. Thus, evaluating $h_{\mathbf{w}}(x)$ is not always tractable, and computational resources limit the expressiveness of the features that can be used.

For example, a first order Markov sequence model has cliques $\{Y_i, Y_{i+1}, X\}$ to score features depending on emissions $\{X, Y_i\}$ and transitions $\{Y_i, Y_{i+1}\}$:

$$\mathbf{w}^\top \mathbf{f}(x, y) = \sum_{i=1}^{\ell} \mathbf{w}^\top \mathbf{f}_i(x, y_i, y_{i+1}). \tag{3}$$

For the first order model, evaluating (1) requires $O(K^2\ell)$ time using the Viterbi decoding algorithm. For an order-d Markov model, with cliques over $\{y_i, \ldots, y_{i+d}, x\}$, inference requires $O(k^{d+1}\ell)$ time.

Several structured prediction methods have been proposed for learning $\mathbf{w}$, such as conditional random fields (Lafferty et al., 2001), structured perceptron (Collins, 2002), hidden Markov support vector machines (Altun et al., 2003) and max-margin Markov networks (Taskar et al., 2003). Our focus here is instead on learning a cascade of increasingly complex models in order to efficiently construct models that would otherwise be intractable.

3.1 Cascaded inference with max-marginals

We will discuss how to learn a cascade of models in Section 4, but first we describe the inference procedure. The basic notion behind our approach is very simple: at each level of the cascade, we receive as input a set of possible clique assignments corresponding to the cliques of the current level. Each level further filters this set of clique assignments and generates a set of possible clique assignments which are passed as input to the next level. Note that each level is able to prune further than the preceding level because it can consider higher-order interactions. At the end of the cascade, the most complex model chooses a single prediction as usual, but it needs only consider the clique assignments that have not already been pruned. Finally, pruning is done using clique max-marginals, for reasons which we describe later in this section.

An example of a cascade for sequential prediction using Markov models is shown is Figure 1. A d-order Markov model has maximal cliques $\{X, Y_i, Y_{i+1}, \ldots, Y_{i+d}\}$. We

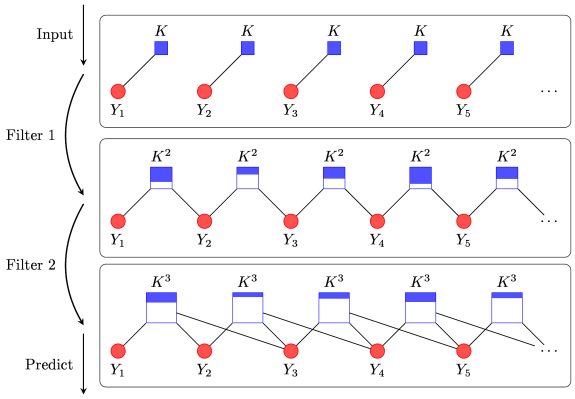

Figure 1: Schematic of a structured prediction cascade using unigram, bigram, and trigram Markov sequence models. Models are represented as factor graphs; as order increases, the state space grows, but the size of the filtered space remains small (filled area).

can consider a cascade of sequence models of increasing order as a set of bigram models where the state space is increasing exponentially by a factor of K from one model to the next. Given a list of valid assignments $\mathcal{V}^i$ in a d-order model, we can generate an expanded list of valid assignments $\mathcal{V}^{i+1}$ for a $(d+1)$-order model by concatenating the valid d-grams with all possible additional states.

More generally, for a given input x, for each maximal clique $c \in \mathcal{C}$ in the current level of the cascade, we have a list of valid assignments $\mathcal{V}_c \subseteq \mathcal{Y}_c$. Inference is performed only over the set of valid clique assignments $\mathcal{V} = \bigcup_{c \in \mathcal{C}} \mathcal{V}_c$. Using the filtering model $\mathbf{w}$ of the current cascade level, the clique assignments in $\mathcal{V}_c$ are scored by their max-marginals, and a threshold t is chosen to prune any $y_c \in \mathcal{V}_c$ with a score less than the threshold. The sets $\mathcal{V}$ are then passed to the next level of the cascade, where higher-order cliques consistent with unpruned cliques are constructed. If we can eliminate at least a fraction of the entries in $\mathcal{V}$ on each round of the cascade, then $|\mathcal{V}|$ decreases exponentially fast and the overall efficiency of inference is controlled.

Thus, to define the cascade, we need to define: (1) the set of models to use in the cascade, and (2) a procedure to choose a threshold t. In the remainder of this section we discuss our approach to these decisions.

First, to define a set of models for the cascade, we require only that the sets of cliques of the models form a nesting sequence. The cliques of the models must satisfy the relation,

$$\mathcal{C}^1 \subseteq \mathcal{C}^2 \subseteq \cdots \subseteq \mathcal{C}^d, \tag{4}$$

where $\mathcal{C}^i$ is the set of cliques of the i'th model of the cascade. In other words, every clique assignment y_c of the i'th model is contained in at least one clique assignment y'_c of the $(i+1)$'th model. This property allows for the use of increasingly complex models as the depth of the cascade increases. As long as (4) holds, a simple and intuitive mapping from the set of valid cliques of the i'th model $\mathcal{V}^i$ to a corresponding set of valid cliques of the $(i+1)$'th model, $\mathcal{V}^i$:

$$\mathcal{V}^{i+1}_c = \{y_c \in \mathcal{Y}_c \mid \forall c' \in \mathcal{C}^i, c' \subseteq c, y_{c'} \in \mathcal{V}^i_{c'}\} \tag{5}$$

This is the set of clique assignments y_c in the $(i+1)$'th model for which all consistent clique assignments $y_{c'}$ for subcliques $c' \in \mathcal{C}^i$ have not been pruned by the i'th model.

In order to filter clique assignments, we use their *max-marginals*. We introduce the shorthand $\theta_x(y) = \mathbf{w}^\top \mathbf{f}(x, y)$ for the cumulative score of an output y, and define the max marginal $\theta^\star_x(y_c)$ as follows:

$$\theta^\star_x(y_c) \triangleq \max_{y' \in \mathcal{Y}} \{\theta_x(y') : y'_c = y_c\}.$$

Computing max-marginals can be achieved using the same dynamic programming inference procedures in factor graphs as would be used to evaluate (1). Most importantly, max marginals satisfy the following simple property:

Lemma 1 (Safe Filtering)**.** *If $\theta_x(y) > \theta^\star_x(y'_c)$ for some c, then $y_c \neq y'_c$.*

Lemma 1 states that if the score of the true label y is greater than the max marginal of a clique assignment y'_c, then that clique assignment y'_c must be inconsistent with the truth y. The lemma follows from the fact that the max marginal of any clique assignment y_c consistent with y is at least the score of y.

A consequence of Lemma 1 is that on training data, a sufficient condition for the target output y not to be pruned is that the threshold t is lower than the score $\theta_x(y)$. Note that a similar condition does not generally hold for standard sum-product marginals of a CRF (where $p(y|x) \propto e^{\theta_x(y)}$), which motivates our use of max-marginals.

The next component of the inference procedure is choosing a threshold t for a given input x. Note that the threshold cannot be defined as a single global value but should instead depend strongly on the input x and $\theta_x(\cdot)$ since scores are on different scales. We also have the constraint that computing a threshold function must be fast enough such that sequentially computing scores and thresholds for multiple models in the cascade does not adversely effect the efficiency of the whole procedure. One might choose a quantile function to consistently eliminate a desired proportion of the max marginals for each example. However, quantile functions are discontinuous in the score function, and we instead approximate a quantile threshold

with a more efficient *mean-max* threshold function, defined as a convex combination of the mean of the max marginals and the maximum score $\theta_x^\star = \max_y \theta_x(y)$,

$$t_x(\alpha) = \alpha\theta_x^\star + (1-\alpha)\frac{1}{|\mathcal{V}|}\sum_{c\in\mathcal{C}, y_c\in V_c}\theta_x^\star(y_c). \quad (6)$$

Choosing a mean-max threshold is therefore choosing $\alpha \in [0,1)$. Note that $t_x(\alpha)$ is a convex function of $\theta_x(\cdot)$ (in fact, piece-wise linear), which combined with Lemma 1 will be important for learning the filtering models and analyzing their generalization. In our experiments, we found that the distribution of max marginals was well centered around the mean, so that choosing $\alpha \approx 0$ resulted in $\approx 50\%$ of max marginals being eliminated on average. As α approaches 1, the number of max marginals eliminated rapidly approaches 100%. We used cross-validation to determine the optimal α in our experiments (section 6).

4 Learning the cascade

We now turn to the problem of finding the best parameters $\mathbf{w}$ and the corresponding best tuning of the threshold α for each level of the cascade. When learning a cascade, we have two competing objectives that we must trade off:

- *Accuracy:* Minimize the number of errors incurred by each level of the cascade to ensure an accurate inference process in subsequent models.
- *Efficiency:* Maximize the number of filtered max marginals at each level in the cascade to ensure an efficient inference process in subsequent models.

We quantify this trade-off by defining two loss functions. We define the *filtering loss* $\mathcal{L}_f$ to be a 0-1 loss indicating a mistakenly eliminated correct assignment. The *efficiency loss* $\mathcal{L}_e$ is the proportion of unfiltered clique assignments.

Definition 1 (Filtering loss)**.** *Let θ_x be the scoring function in the current level of the cascade. A filtering error occurs when a max-marginal of a clique assignment of the correct output y is pruned. We define filtering loss as $\mathcal{L}_f(y, \theta_x) = \mathbf{1}\left[\theta_x(y) \le t_x(\alpha)\right]$.*

Definition 2 (Efficiency loss)**.** *Let θ_x be defined as above. The efficiency loss is the proportion of unpruned clique assignments $\mathcal{L}_e(y, \theta_x) = \frac{1}{|\mathcal{V}|}\sum_{c\in C, y_c\in\mathcal{V}_c}\mathbf{1}\left[\theta_x^\star(y_c) \ge t_x(\alpha)\right]$.*

Note that we can trivially minimize either of these at the expense of maximizing the other. If we set $(\mathbf{w}, \alpha)$ to achieve a minimal threshold such that no assignments are ever filtered, then $\mathcal{L}_f = 0$ and $\mathcal{L}_e = 1$. Alternatively, if we choose a threshold to filter every assignment, then $\mathcal{L}_f = 1$ while $\mathcal{L}_e = 0$. To learn a cascade of practical value, we can minimize one loss while constraining the other below a threshold ϵ. Since the ultimate goal of the cascade is accurate classification, we focus on the problem of minimizing efficiency loss while constraining the filtering loss to be below a desired tolerance ϵ.

We can express the cascade learning objective as a joint optimization over $\mathbf{w}$ and α:

$$\min_{\mathbf{w},\alpha}\ \mathbb{E}\left[\mathcal{L}_e(Y, \theta_X)\right] \text{ s.t. } \mathbb{E}\left[\mathcal{L}_f(Y, \theta_X)\right] \le \epsilon, \quad (7)$$

We solve this problem with a two-step procedure. First, we define a convex upper-bound on the filter error $\mathcal{L}_f$, making the problem of minimizing $\mathcal{L}_f$ convex in $\mathbf{w}$ (given α). We learn $\mathbf{w}$ to minimize filter error for several settings of α (thus controlling filtering efficiency). Second, given $\mathbf{w}$, we optimize the objective (7) over α directly, using estimates of $\mathcal{L}_f$ and $\mathcal{L}_e$ computed on a held-out development set. In section 5 we present a theorem bounding the deviation of our estimates of the efficiency and filtering loss from the expectation of these losses.

To learn one level of the structured cascade model $\mathbf{w}$ for a fixed α, we pose the following convex margin optimization problem:

$$\mathbf{SC}: \quad \inf_{\mathbf{w}} \quad \frac{\lambda}{2}||\mathbf{w}||^2 + \frac{1}{n}\sum_i H(\mathbf{w}; (x^i, y^i)), \quad (8)$$

where H is a convex upper bound on the filter loss $\mathcal{L}_f$,

$$H(\mathbf{w}; (x^i, y^i)) = \max\{0, \ell + t_{x^i}(\alpha) - \mathbf{w}^\top\mathbf{f}(x^i, y^i)\}.$$

The upper-bound H is a hinge loss measuring the margin between the filter threshold $t_{x^i}(\alpha)$ and the score of the truth $\mathbf{w}^\top\mathbf{f}(x^i, y^i)$; the loss is zero if the truth scores above the threshold by margin ℓ (in practice, the length ℓ can vary by example). We solve (8) using stochastic sub-gradient descent. Given a sample (x, y), we apply the following update if $H(\mathbf{w}; (x, y))$ (i.e., the sub-gradient) is non-zero:

$$\begin{aligned}\mathbf{w}' \leftarrow (1-\lambda)\mathbf{w} + \eta\mathbf{f}(x, y) - \eta\alpha\mathbf{f}(x, y^\star) \\ - \eta(1-\alpha)\frac{1}{|\mathcal{V}|}\sum_{c\in\mathcal{C}, y_c}\mathbf{f}(x, y^\star(y_c)).\end{aligned} \quad (9)$$

Above, η is a learning rate parameter, $y^\star = \text{argmax}_{y'}\,\theta_x(y')$ and $y^\star(y_c) = \text{argmax}_{y':y_c=y'_c}\,\theta_x(y')$. The key distinguishing feature of the this update as compared to structured perceptron is that it subtracts features included in all max marginal assignments $y^\star(y_c)$.

Note that because (8) is λ-strongly convex, if we chose $\eta_t = 1/(\lambda t)$ and add a projection step to keep $\mathbf{w}$ in a

closed set, the update would correspond to the Pegasos update with convergence guarantees of $\tilde{O}(1/\epsilon)$ iterations for ϵ-accurate solutions (Shalev-Shwartz, Singer, and Srebro, 2007).

5 Generalization Analysis

In this section, we give generalization bounds on the filtering and efficiency loss functions for a single level of a cascade. These bounds depend on Lipschitz dominating cost functions ϕ_f and ϕ_e that upper bound $\mathcal{L}_f$ and $\mathcal{L}_e$. To formulate these functions, we define the the scoring function $\theta_x : \mathcal{X} \mapsto \mathbb{R}^m$ where $m = \sum_{c \in \mathcal{C}} |\mathcal{Y}_c|$ to be the set of scores for all possible clique assignments, $\theta_x = \{\mathbf{w}^\top \mathbf{f}_c(x, y_c) \mid c \in \mathcal{C}, y_c \in \mathcal{Y}_c\}$. Thus, given θ_x, the score $\theta_x(y)$ can be computed as the inner product $\langle \theta_x, y \rangle$, where we treat y as a m-vector of indicators where $y_i = 1$ if the i'th clique assignment appears in y. Finally, we define the auxiliary function ϕ to be the difference between the score of output y and the threshold as $\phi(y, \theta_x) = \theta_x(y) - t_x(\alpha)$. We now state the main result of this section.

Theorem 1. *Let θ_x, $\mathcal{L}_e$, $\mathcal{L}_f$, and ϕ be defined as above. Let Θ be the class of all scoring functions θ_X with $||\mathbf{w}||_2 \leq B$, the total number of cliques ℓ, and $||\mathbf{f}(x, y_c)||_2 \leq 1$ for all x and y_c. Define the dominating cost functions $\phi_f(y, \theta_X) = r_\gamma(\phi(y, \theta_X))$ and $\phi_e(y, \theta_X) = \frac{1}{m} \sum_{c \in \mathcal{C}, y_c} r_\gamma(\phi(y^\star(y_c), -\theta_X))$, where $r_\gamma(\cdot)$ is the ramp function with slope γ. Then for any integer n and any $0 < \delta < 1$ with probability $1 - \delta$ over samples of size n, every $\theta_X \in \Theta$ and $\alpha \in [0, 1]$ satisfies:*

$$\mathbb{E}\left[\mathcal{L}_f(Y, \theta_X)\right] \leq \hat{\mathbb{E}}\left[\phi_f(Y, \theta_X)\right] + O\left(\frac{m\sqrt{\ell}B}{\gamma\sqrt{n}}\right) + \sqrt{\frac{8\ln(2/\delta)}{n}}, \tag{10}$$

where $\hat{\mathbb{E}}$ is the empirical expectation with respect to training data. Furthermore, (10) *holds with $\mathcal{L}_f$ and ϕ_f replaced by $\mathcal{L}_e$ and ϕ_e.*

This theorem relies on the general bound given in Bartlett and Mendelson (2002), the properties of Rademacher and Gaussian complexities (also in Bartlett and Mendelson (2002)), and the following lemma:

Lemma 2. *$\phi_f(y, \cdot)$ and $\phi_e(y, \cdot)$ are Lipschitz (with respect to Euclidean distance on $\mathbb{R}^m$) with constant $\sqrt{2\ell}/\gamma$.*

A detailed proof of Theorem 1 and Lemma 2 is given in the appendix.

Theorem 1 provides theoretical justification for the definitions of the loss functions $\mathcal{L}_e$ and $\mathcal{L}_f$ and the structured cascade objective; if we observe a highly accurate and efficient filtering model $(\mathbf{w}, \alpha)$ on a finite sample of training data, it is likely that the performance of the model on unseen test data will not be too much worse as n gets large. Theorem 1 is the first theoretical guarantee on the generalization of *accuracy* and *efficiency* of a structured filtering model.

6 Experiments

Handwriting Recognition. We first evaluated the accuracy of the cascade using the handwriting recognition dataset from Taskar et al. (2003). This dataset consists of 6877 handwritten words, with avereage length of ∼8 characters, from 150 human subjects, from the data set collected by Kassel (1995). Each word was segmented into characters, each character was rasterized into an image of 16 by 8 binary pixels. The dataset is divided into 10 folds; we used 9 folds for training and a single withheld for testing (note that Taskar et al. (2003) used 9 folds for testing and 1 for training due to computational limitations, so our results are not directly comparable). Results are averaged across all 10 folds.

Our objective was to measure the improvement in predictive accuracy as higher order models were incorporated into the cascade. The final cascade consisted of four Markov models of increasing order. Pixels are used as features for the lowest order model, and 2,3, and 4-grams of letters are the features for the three higher order models, respectively. The maximum filter loss threshold ϵ was set to 1%, 2%, and 4% (this avoids over-penalizing higher order models), and we trained structured cascades (SC) using α's from the candidate set $\{0, 0.25, 0.5\}$. To simplify training, we fixed $\eta = 1$, $\lambda = 0$, and used an early-stopping procedure to choose $\mathbf{w}$ that achieved optimal tradeoff according to (7).

Results are summarized in Table 1. We found that the use of higher order models dramatically increased accuracy of the predictions, raising accuracy at the character above 90% and more than tripling the word-level accuracy. Furthermore, the cascade reduced the search space of the 4-gram model by 5 orders of magnitude, while incurring only 3.41% filtering error.

Part-of-Speech Tagging. We next evaluated our approach on several part of speech (POS) tagging tasks. Our objective was to rigorously compare the efficacy of our approach to alternative methods on a problem while reproducing well-established benchmark prediction accuracy. The goal of POS tagging is to assign a label to each word in a sentence. We used three different languages in our experiments: English, Portuguese and Bulgarian. For English, we used vol-

umes 0-17 of the Wall Street Journal portion of the Penn TreeBank (Marcus et al., 1993) for training, volumes 18-20 for development, and volumes 21-24 for testing. We used the Bulgarian BulTreeBank (Simov et al., 2002) and the Bosque subset of the Portuguese Floresta Sinta(c)tica (Afonso et al., 2002) for Bulgarian and Portuguese datasets, respectively.

For these experiments, we focused on comparing the efficiency of our approach to the efficiency of several alternative approaches. We again used a set of Markov models of increasing order; however, to increase filtering efficiency, we computed max marginals over subcliques representing $(d-1)$ order state assignments rather than d-order cliques representing transitions. Thus, the bigram model filters unigrams and the trigram model filters bigrams, etc. Although we ran the cascades up to 5-gram models, peak accuracy was reached with trigrams on the POS task.

We compared our SC method to two baselines: the standard structured perceptron (SP) and a maximum a posteriori CRF. All methods used the same standard set of binary features: namely, a feature each (word,tag) pair $\mathbf{1}\left[X_t = x_t, Y_t = y_t\right]$ and feature for each d-gram in the model. For the baseline methods, we trained to minimize classification error (for SP) or log-loss (for CRF) and then chose $\alpha \in [0,1]$ to achieve minimum $\mathcal{L}_e$ subject to $\mathcal{L}_f \leq \epsilon$ on the development set. For the CRF, we computed sum-product marginals $P(y_c|x) = \sum_{y':y'_c=y_c} P(y'|x)$, and used the threshold $t_x(\alpha) = \alpha$ to eliminate all y_c such that $P(y_c|x) \leq \alpha$. For all algorithms, we used grid search over several values of η and λ, in addition to early stopping, to choose the optimal $\mathbf{w}$ based on the development set. For SC training, we considered initial α's from the candidate set $\{0, 0.2, 0.4, 0.6, 0.8\}$.

We first evaluated our approach on the WSJ dataset. To ensure accuracy of the final predictor, we set a strict threshold of $\epsilon = 0.01\%$. All methods were trained using a structured perceptron for the final classifier. The results are summarized in Table 2. SC was compared to CRF, an unfiltered SP model (Full), and a heuristic baseline in which only POS tags associated with a given word in the training set were searched during inference (Tags). SC was two orders of magnitude faster than the full model in practice, with the search space reduced to only ≈ 4 states per position in inference (roughly the complexity of a greedy approximate beam search with 4 beams.) SC also outperformed CRF by a factor of 2.6 and the heuristic by a factor of 6.8. Note that because of the trade-off between accuracy and efficiency, CRF suffered less filter loss relative to SC due to pruning less aggressively, although neither method suffered enough filtering loss to affect the accuracy of the final classifier. Finally, training the full trigram POS tagger with exact inference is extremely slow and took several days to train; training the SC cascade took only a few hours, despite the fact that more models were trained in total.

Model Order:	**1**	**2**	**3**	**4**
Accuracy, Char. (%)	77.44	85.69	87.95	92.25
Accuracy, Word (%)	26.65	49.44	73.83	84.46
Filter Loss (%)	0.56	0.99	3.41	—
Avg. Num n-grams	26.0	123.8	88.6	5.4

Table 1: Summary of handwriting recognition results. For each level of the cascade, we computed prediction accuracy (at character and word levels) using a standard voting perceptron algorithm as well as the filtering loss and average number of unfiltered n-grams per position for the SC on the test set.

Model:	**Full**	**SC**	**CRF**	**Tags**
Accuracy (%)	96.83	96.82	96.84	—
Filter loss (%)	0	0.121	**0.024**	0.118
Test Time (ms)	173.28	**1.56**	4.16	10.6
Avg. Num States	1935.7	**3.93**	11.845	95.39

Table 2: Summary of WSJ Results. Accuracy is the accuracy of the final trigram model. Filter loss is the number of incorrectly pruned bigrams at the end of the cascade. The last row is the average number of states considered at each position in the test set.

We next investigated the efficiency vs. filtering accuracy trade-off of SC compared to the SP and CRF baselines on all three languages. For each of the three languages, we generated 10 different training sets from 40% of the full training datasets. Randomization was taken with the same 10 seeds for each dataset/algorithm pair. For all methods, we trained bigram models under two conditions: first, as the initial step of a cascade (e.g., no prior filtering), and second, as the second step of a cascade after initial filtering by the SC algorithm with $\epsilon = 0.05\%$, and analyzed the resulting trade-off between efficiency and accuracy.

The results are presented in Figure 2. The figures were generated by computing for each ϵ along the x-axis the corresponding *test* efficiency $\mathcal{L}_e$ when the constraint on filter loss ($\mathcal{L}_f \leq \epsilon$) is enforced using *development* set data. Points for which the constraint could not be enforced (even with $\alpha = 0$) are not shown. SC handily beats the competitors in both the filtered and unfiltered case. Note that in the unfiltered condition, the CRF is unable to achieve significant pruning efficiency for any ϵ, while the SP cannot achieve filtering accuracy for small ϵ. However, because SC and SP become equivalent as α approaches 1, we observe that the performance of SC and SP converge as ϵ increases.

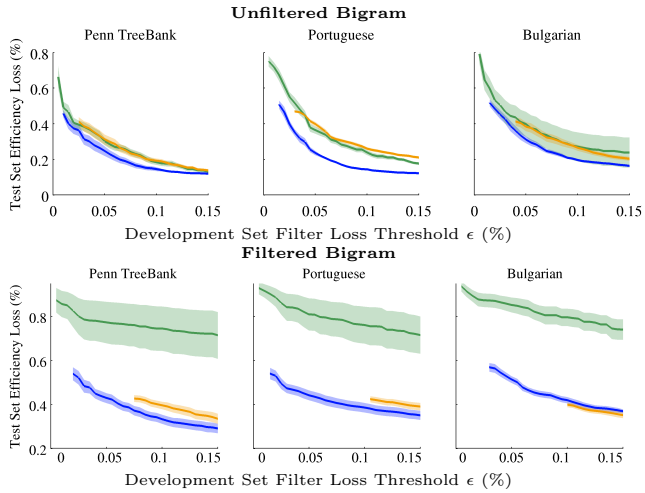

Figure 2: Efficiency vs. accuracy trade-off for unfiltered (top) and filtered (bottom) bigram models on POS tagging task. Blue (darkest) is SC, green (lighter) is CRF, and orange is SP (lightest). Shaded region shows standard error over 10 different random training sets. The y-axis show the test set efficiency when fitting to enforce the constraint $\mathcal{L}_f \leq \epsilon$ (x-axis) on development data.

7 Conclusion

We presented structured prediction cascades: an efficient, principled method of tackling the complexity of structured prediction problems. We proposed and theoretically analyzed a novel loss function that attempts to balance accuracy and efficiency of the cascade. Using a simple sub-gradient optimization method, we evaluated our method on several large-scale datasets. On the handwriting recognition task, we found that the cascade reduced the state space of a fourth-order Markov model by five orders of magnitude, allowing the use of higher order features that tripled the word-level accuracy of predictions. On the POS task, the cascade increased inference efficiency of trigram models 100× relative to unfiltered inference without negatively impacting the accuracy of the final predictor.

While we used sequence models to illustrate the cascades, the methods described here could be applied to other sets of nested models. For example, graphical models used for articulated object detection in vision have a natural coarse-to-fine scale hierarchy. Temporal models with factored state (e.g., dynamic Bayes nets or factorial CRFs) are another natural candidate for cascaded learning and inference. Our approach could also be extended to incorporate intractable models through the use of approximate inference techniques. Theoretical analysis of filtering accuracy in the case of approximate inference is an interesting open question.

A Proof of Theorem 1

Assume Lemma 2 holds (that ϕ_f and ϕ_e are Lipschitz with constant $\sqrt{2\ell}/\gamma$) as shown below. We now reproduce the following four facts from Bartlett and Mendelson (2002) that we require to prove the theorem.

Theorem 2 (Bartlett and Mendelson, 2002)**.** *Consider a loss function $\mathcal{L}$ and a dominating cost function ϕ such that $\mathcal{L}(y,x) \leq \phi(y,x)$. Let $F : \mathcal{X} \mapsto \mathcal{A}$ be a class of functions. Then for any integer n and any $0 < \delta < 1$, with probability $1-\delta$ over samples of length n, every f in F satisfies*

$$\mathbb{E}\mathcal{L}(Y, f(X)) \leq \hat{\mathbb{E}}_n \phi(Y, f(X)) + R_n(\tilde{\phi} \circ F) + \sqrt{\frac{8\ln(2/\delta)}{n}}, \tag{11}$$

where $\tilde{\phi} \circ F$ is a centered composition of ϕ with $f \in F$, $\tilde{\phi} \circ f = \phi(y, f(X)) - \phi(y, 0)$.

Furthermore, there are absolute constants c and C such that for every class F and every integer n,

$$cR_n(F) \leq G_n(F) \leq C \ln n R_n(F). \tag{12}$$

Let $\mathcal{A} = \mathbb{R}^m$ and $F : \mathcal{X} \to \mathcal{A}$ be a class of functions that is the direct sum of real-valued classes $F_1, \ldots, F_m$. Then, for every integer n and every sample $(X_1, Y_1), \ldots, (X_n, Y_n)$,

$$\hat{G}_n(\tilde{\phi} \circ F) \leq 2L \sum_{i=1}^{m} \hat{G}_n(F_i). \tag{13}$$

Let $F = \{x \mapsto \mathbf{w}^\top \mathbf{f}(x, \cdot) \mid ||\mathbf{w}||_2 \leq B, ||\mathbf{f}(x,\cdot)||_2 \leq 1\}$. Then,

$$\hat{G}_n(F) \leq \frac{2B}{\sqrt{n}}. \tag{14}$$

Proof of Theorem 1. Let $\mathcal{A} = \mathbb{R}^m$ and $F = \Theta_X$. By (12), we have that $R_n(\tilde{\phi}_f \circ F) = O(G_n(\tilde{\phi}_f \circ F))$. Since $\tilde{\phi}$ passes through the origin, then by (13), $G_n(\tilde{\phi}_f \circ F) = O(2L(\tilde{\phi}_f) \sum_i G_n(f_i)) = O(mL(\tilde{\phi}_f) G_n(\mathcal{H}))$. With Lemma 2 and $L(\tilde{\phi}_f) = L(\phi_f)$, we then have that $R_n(\tilde{\phi}_f \circ F) = O(m\gamma^{-1}\sqrt{l} G_n(F_i))$, where F_i is a linear function scoring the i'th clique assignment. Thus the results follows from (14) and (11). Finally, because $L(\phi_f) = L(\phi_e)$, the same results applies to $\mathcal{L}_e$ as well. □

A.1 Proof of Lemma 2

To prove Lemma 2, we first bound the slope of the difference $\phi(y, \theta_x)$. We observe that we can consider the scores of output y as the inner product $\langle y, \theta_x \rangle$, where we treat y as a m-vector of indicators and $y_i = 1$ if the i'th clique assignment appears in y.

Lemma 3. *Let* $f(\theta_x) = \langle y, \theta_x \rangle - \max_{y'} \langle y', \theta_x \rangle$. *Then* $f(u) - f(v) \leq \sqrt{2\ell}||u - v||_2$.

Proof. Let $y_u = \text{argmax}_{y'} \langle y', u \rangle$ and $y_v = \text{argmax}_{y'} \langle y', v \rangle$. Then we have,

$$\begin{aligned} f(u) - f(v) &= \langle y, u \rangle - \langle y_u, u \rangle + \langle y_v, v \rangle - \langle y, v \rangle \\ &= \langle y_v - y, v \rangle + \langle y - u_u, u \rangle + \langle y_v, u \rangle - \langle y_v, u \rangle \\ &= \langle y_v - y, v - u \rangle + \langle u, y_v - y_u \rangle \\ &\leq \langle y_v - y, v - u \rangle \\ &\leq ||y_v - y||_2 ||u - v||_2 \leq \sqrt{2\ell}||u - v||_2. \end{aligned}$$

The last two steps follow from the fact that y_u maximizes $\langle y_u, u \rangle$ (so $\langle u, y_v - y_u \rangle$ is negative), application of Cauchy-Schwarz, and from the fact that there are at most ℓ cliques appear, each of which can contribute a single non-zero entry in y or y_v. □

Lemma 4. *Let* $f'(\theta_x) = \langle y, \theta_x \rangle - \frac{1}{m} \sum_{c \in \mathcal{C}, y_c} \max_{y' : y'_c = y_c} \langle y', \theta_x \rangle$. *Then* $f(u) - f(v) \leq \sqrt{2\ell}||u - v||_2$.

Proof. Let $y_{ui} = \text{argmax}_{y' : y'_c = y_c} \langle y', u \rangle$ for the i'th clique assignment y_c, and y_{vi} the same for v. Then we have,

$$\begin{aligned} f'(u) - f'(v) &= \frac{1}{m} \sum_{i=1}^{m} \langle y, u \rangle - \langle y_{ui}, u \rangle + \langle y_{vi}, v \rangle - \langle y, v \rangle \\ &\leq \frac{1}{m} \sum_{i=1}^{m} \langle y_{vi} - y, v - u \rangle \\ &\leq \frac{1}{m} \sum_{i=1}^{m} \sqrt{2\ell}||u - v||_2 = \sqrt{2\ell}||u - v||_2. \end{aligned}$$

Here we have condensed the same argument used to prove the previous lemma. □

Lemma 5. *Let* $g(\theta_x) = \langle y, \theta_x \rangle - t_x(\alpha)$. *Then* $g(u) - g(v) \leq \sqrt{2\ell}||u - v||_2$.

Proof. Plugging in the definition of $t_x(\alpha)$, we see that $g(\theta_x) = \alpha f(\theta_x) + (1-\alpha) f'(\theta_x)$. Therefore from the previous two lemmas we have that $g(u) - g(v) = \alpha(f(u) - f(v)) + (1 - \alpha)(f'(u) - f'(v)) \leq \sqrt{2\ell}||u - v||_2$. □

From Lemma 5, we see that $\phi(y, \cdot) = g(\cdot)$ is Lipschitz with constant $\sqrt{2\ell}$. We can now show that ϕ_f and ϕ_e are Lipschitz continuous with constant $\sqrt{2\ell}/\gamma$. Let $L(\cdot)$ denote the Lipschitz constant. Then $L(\phi_t) = L(r_\gamma) \cdot L(\phi(y, \cdot)) \leq \sqrt{2\ell}/\gamma$.

To show $L(\phi_e)$ requires more bookkeeping because we must bound $\phi(y^\star(y_c), \theta_x)$. We can prove equivalent lemmas to lemmas 3, 4 and 5 where we substitute $\langle y, \theta_x \rangle$ with $\max_{y' : y'_c = y_c} \langle y', \theta_x \rangle$, and thus show that $L(\phi(y^\star(y_c), \cdot)) \leq \sqrt{2\ell}$. Therefore, $L(\phi_e) = \frac{1}{m} \sum_{c \in \mathcal{C}, y_c} L(r_\gamma) \cdot L(\phi(y^\star(y_c), \cdot)) \leq \sqrt{2\ell}/\gamma$, as desired.

Acknowledgements

This work was partially supported by NSF Grant 0803256.

References

S. Afonso, E. Bick, R. Haber, and D. Santos. Floresta Sinta(c)tica: a treebank for Portuguese. In *Proc. LREC*, 2002.

Y. Altun, I. Tsochantaridis, and T. Hofmann. Hidden Markov support vector machines. In *Proc. ICML*, 2003.

P. L. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

X. Carreras, M. Collins, and T. Koo. TAG, dynamic programming, and the perceptron for efficient, feature-rich parsing. In *Proc. CoNLL*, 2008.

E. Charniak. A maximum-entropy-inspired parser. In *Proc. NAACL*, 2000.

M. Collins. Discriminative training methods for hidden markov models: theory and experiments with perceptron algorithms. In *Proc. EMNLP*, 2002.

G. Druck, M. A. Amherst, M. Narasimhan, W. A. Redmond, and P. Viola. Learning A* underestimates: Using inference to guide inference. In *Proc. AISTATS*, 2007.

R. Kassel. *A Comparison of Approaches to On-line Handwritten Character Recognition*. PhD thesis, Massachusetts Institute of Technology, 1995.

J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. ICML*, 2001.

M. Marcus, S. Santorini, and M. Marcinkiewicz. Building a large annotated corpus of english: the penn treebank. *Computational Linguistics*, 19(2):313–330, 1993.

C. Pal, C. Sutton, and A. McCallum. Sparse forward-backward using minimum divergence beams for fast training of CRFs. In *Proc. ICASP*, 2006.

S. Petrov. *Coarse-to-Fine Natural Language Processing*. PhD thesis, University of California at Bekeley, 2009.

A. Ratnaparkhi. A maximum entropy model for part-of-speech tagging. In *Proc. EMNLP*, 1996.

S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal estimated sub-gradient SOlver for SVM. In *Proc. ICML*, 2007.

K. Simov, P. Osenova, M. Slavcheva, S. Kolkovska, E. Balabanova, D. Doikoff, K. Ivanova, A. Simov, E. Simov, and M. Kouylekov. Building a linguistically interpreted corpus of bulgarian: the bultreebank. In *Proc. LREC*, 2002.

B. Taskar, C. Guestrin, and D. Koller. Max margin Markov networks. In *Proc. NIPS*, 2003.

P. Viola and M. Jones. Robust real-time object detection. *International Journal of Computer Vision*, 57(2):137–154, 2002.