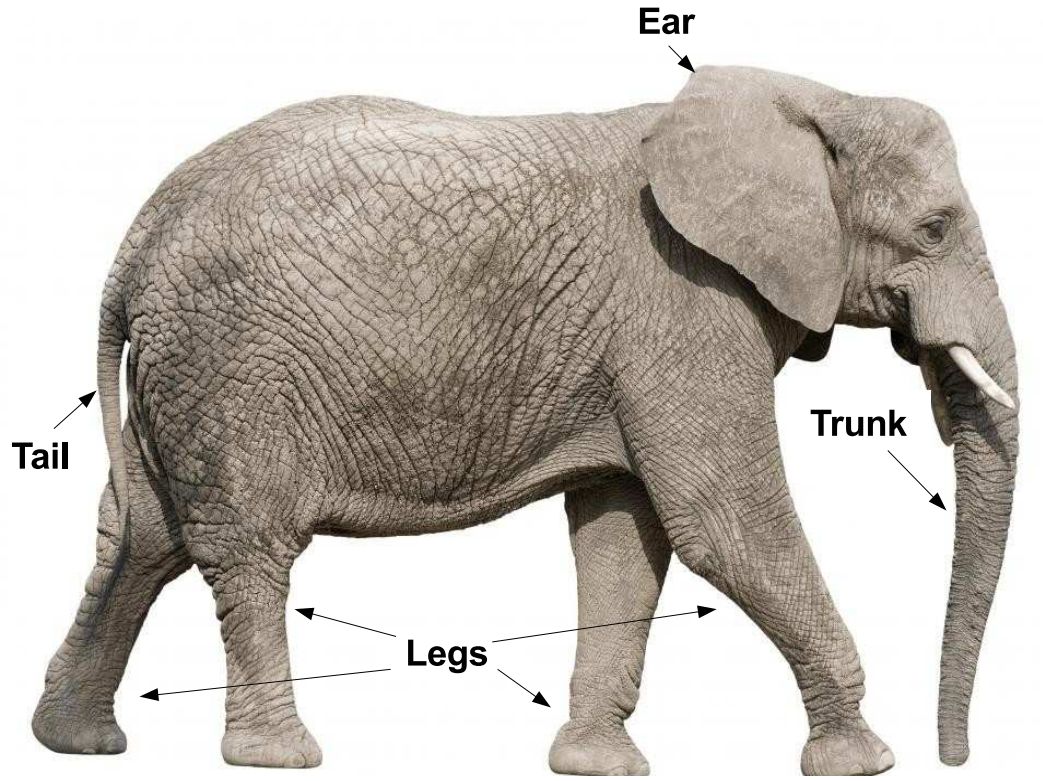
Ensemble Methods for Structured Prediction

Vitaly Kuznetsov[1]
Joint work with
Corinna Cortes[2] and Mehryar Mohri[1,2]

[1]Courant Institute of Mathematical Sciences, New York University

[2]Google Research, New York

Structured Prediction



Structured Prediction

Graphemes-to-phonemes task:

- **Input:** sequence of graphemes, e.g.

  ensemble algorithm

- **Output:** sequence of phonemes, e.g.

  än-ˈsäm-bəl ˈal-gə-ri-thəm

Ensemble Methods

- Often significantly improve performance.
- Benefit from favorable learning guarantees.
- Developed primarily for classification and regression tasks.

Ensembles & Structured Prediction

Input:	ensemble	algorithm
Expert 1:	än-'səm-bəl	'al-gȯ-ri-thəm
Expert 2:	ən-'säm-bəl	əl-gə-ri-thəm

Goal: learn to **patch together** predictions of different experts.

Outline

- Learning scenario.
- Prior work.
- Boosting algorithm.
- On-line solution.
- Experiments.

Learning Scenario

- Learner receives a sample $(\mathbf{x}_i, \mathbf{y}_i)_{i=1}^m \in \mathcal{X} \times \mathcal{Y}$.
- $\mathbf{y} \in \mathcal{Y}$ decomposes into $\mathbf{y} = (y^1, \ldots, y^l)$.
- Loss is additive

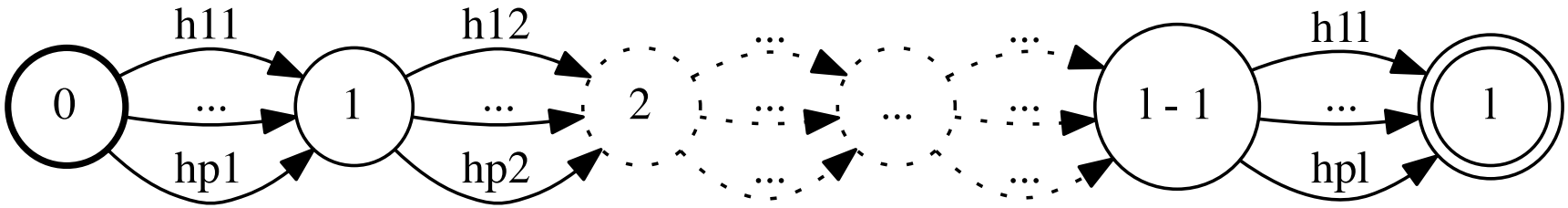
$$L(\mathbf{y}, \widetilde{\mathbf{y}}) = \sum_{k=1}^{l} \ell(y^k, \widetilde{y}^k).$$

- Learner has access to **black box** predictors $h_1, \ldots, h_p : \mathcal{X} \to \mathcal{Y}$.

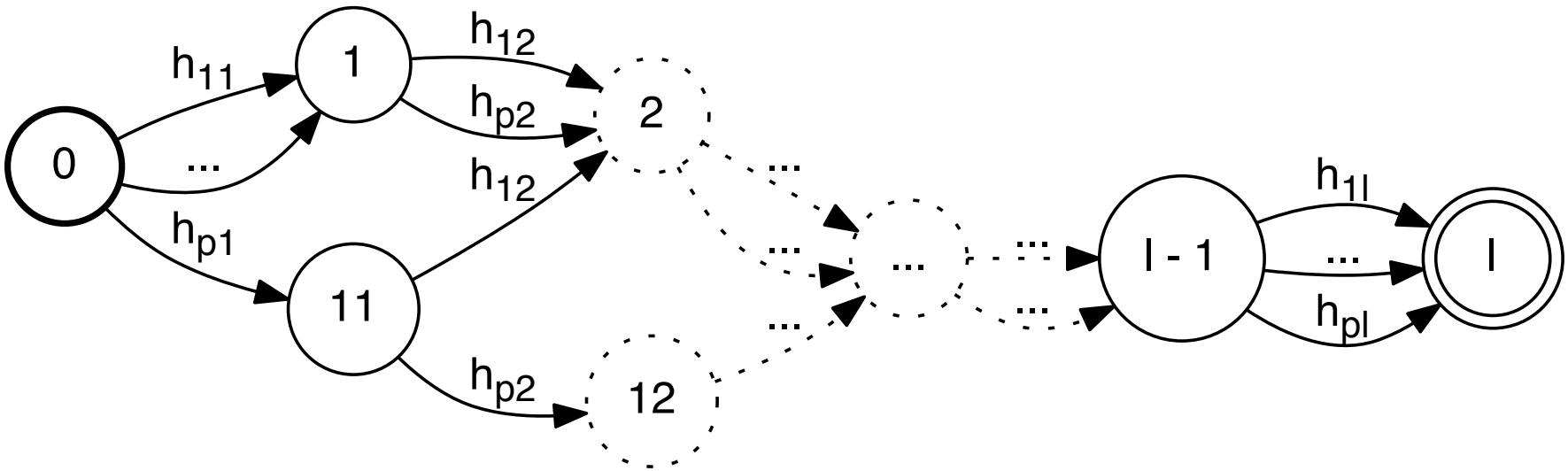
Prior Work

- **Re-ranking techniques:** Collins & Koo, 2005; Huang, 2008.
- **Combinations:** Zeman & Žabokrtský, 2005; Sagae & Lavie, 2006.
- **Scores:** Mohri et. al., 2008; Petrov, 2010; Zhang et. al., 2009.
- **Special experts:** Kocev et. al., 2013; Wang et. al., 2007; Fiscus, 1997.
- **SLE algorithm:** Nguyen & Guo, 2007.

Path Experts.



General Graphs



Boosting Framework

- Learn a scoring function $\widetilde{\mathsf{h}} = \sum_{t=1}^{T} \alpha_t \widetilde{\mathsf{h}}_t$, $\alpha_t > 0$.
- Predict

$$\mathcal{H}_{\text{Boost}}(\mathbf{x}) = \operatorname*{argmax}_{\mathbf{y} \in \mathcal{Y}} \widetilde{\mathsf{h}}(\mathbf{x}, \mathbf{y}).$$

- No restrictions on base scoring functions h_j.
- For black box experts

$$\widetilde{\mathsf{h}}_t(\mathbf{x}, \mathbf{y}) = \sum_{k=1}^{l} \mathbf{1}_{\mathsf{h}_t^k(\mathbf{x}) = y^k}.$$

ESPBoost

Inputs: Sample S; experts $\{h_1, \ldots, h_p\}$.
for $i = 1$ **to** m **and** $k = 1$ **to** l **do**
  $\mathcal{D}_1(i,k) \leftarrow \frac{1}{ml}$
end for
for $t = 1$ **to** T **do**
  $\mathsf{h}_t \leftarrow \operatorname{argmin}_{\mathsf{h}\in\mathsf{H}} \mathbb{E}_{(i,k)\sim\mathcal{D}_t}[\mathbf{1}_{\mathsf{h}^k(\mathbf{x}_i)\neq y_i^k}]$
  $\epsilon_t \leftarrow \mathbb{E}_{(i,k)\sim\mathcal{D}_t}[\mathbf{1}_{\mathsf{h}_t^k(\mathbf{x}_i)\neq y_i^k}]$
  $\alpha_t \leftarrow \frac{1}{2}\log\frac{1-\epsilon_t}{\epsilon_t}$
  $Z_t \leftarrow 2\sqrt{\epsilon_t(1-\epsilon_t)}$
  for $i = 1$ **to** m **and** $k = 1$ **to** l **do**
    $\mathcal{D}_{t+1}(i,k) \leftarrow \frac{\exp(-\alpha_t \rho(\widetilde{\mathsf{h}}_t^k, \mathbf{x}_i, \mathbf{y}_i))\mathcal{D}_t(i,k)}{Z_t}$
  end for
end for

ESPBoost algorithm

- Upper bound on empirical loss:

$$\frac{1}{ml}\sum_{i=1}^{m}\sum_{k=1}^{l}\mathbf{1}_{\mathcal{H}^k_{\mathrm{Boost}}(\mathbf{x}_i)\neq y_i^k}$$
$$\leq \frac{1}{ml}\sum_{i=1}^{m}\sum_{k=1}^{l}\exp\Big(-\sum_{t=1}^{T}\alpha_t\rho(\widetilde{\mathrm{h}}_t^k,\mathbf{x}_i,\mathbf{y}_i)\Big),$$

- $\rho(\widetilde{\mathrm{h}}_t^k,\mathbf{x}_i,\mathbf{y}_i)$ is margin of the $\widetilde{\mathrm{h}}_t$ at position k on example (x_i,y_i).
- ESPBoost algorithm is an application of the coordinate descent to this bound.

Learning guarantees

Theorem

Fix $\rho > 0$. Then, for any $\delta > 0$, with probability at least $1 - \delta$, the following holds for all $\mathcal{H}_{\mathrm{Boost}} \in \mathcal{F}$:

$$\mathop{\mathbb{E}}_{(\mathbf{x},\mathbf{y})\sim\mathcal{D}}[L_{\mathrm{Ham}}(\mathcal{H}_{\mathrm{Boost}}(\mathbf{x}), \mathbf{y})] \leq \widehat{R}_\rho\left(\frac{\widetilde{\mathsf{h}}}{\|\boldsymbol{\alpha}\|_1}\right) + \frac{2}{\rho l}\sum_{k=1}^{l} |\mathcal{Y}_k| \mathfrak{R}_m(H^k) + \sqrt{\frac{\log \frac{l}{\delta}}{2m}},$$

where $\mathfrak{R}_m(H^k)$ denotes the Rademacher complexity of the class of functions $\{\mathbf{x} \mapsto \mathsf{h}_j(\mathbf{x}, y) : j \in [1, p], y \in \mathcal{Y}_k\}$.

Learning guarantees

Theorem

Let $\widetilde{\mathsf{h}}$ *denote the scoring function returned by ESPBoost after* $T \geq 1$ *rounds. Then, for any* $\rho > 0$*, the following inequality holds*

$$\widehat{R}_{\rho}\left(\frac{\widetilde{\mathsf{h}}}{\|\boldsymbol{\alpha}\|_1}\right) \leq 2^T \prod_{t=1}^{T} \sqrt{\epsilon_t^{1-\rho}(1-\epsilon_t)^{1+\rho}}.$$

On-line Algorithm

Two stage procedure:

1. Run on-line learning algorithm to learn a distribution $\mathbf{p}$ over path experts.
2. Convert on-line solution $\mathbf{p}$ to a batch predictor.

Options for on-line algorithms:

Follow-the-Perturbed-Leader (FPL).

Randomized Weighted Majority (RWM).

RWM Algorithm (Littlestone & Warmuth, 1994)

- $\mathbf{p}_0$ is a uniform distribution over paths experts.
- Receive $(\mathbf{x}_t, \mathbf{y}_t)$.
- Update

$$\mathbf{p}_{t+1}(\mathrm{h}) = \frac{\mathbf{p}_t(\mathrm{h})\beta^{L(\mathrm{h}(x_t),\mathbf{y}_t)}}{\sum_{\mathrm{h}'} \mathbf{p}_t(\mathrm{h}')\beta^{L(\mathrm{h}'(x_t),\mathbf{y}_t)}}.$$

- Efficient updates using structure of the problem (Takimoto & Warmuth, 2003).

On-line-to-batch Conversion

- Choose ensemble $\mathcal{P}$ to minimize:

$$\Gamma(\mathcal{P}) = \frac{1}{|\mathcal{P}|} \sum_{\mathbf{p}_t \in \mathcal{P}} \mathop{\mathbb{E}}_{\mathsf{h} \sim p_t} [L(\mathsf{h}(x_t), \mathbf{y}_t)] + M \sqrt{\frac{\log \frac{1}{\delta}}{|\mathcal{P}|}}.$$

- Form ensemble distribution $\mathbf{p} = \frac{1}{|\mathcal{P}|} \sum_{\mathbf{p}_t \in \mathcal{P}} \mathbf{p}_t$.
- Stochastic or voting predictions.

Learning Guarantees

Theorem

For any $\delta > 0$, with probability at least $1 - \delta$ over the choice of the sample $((\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_T, \mathbf{y}_T))$ drawn i.i.d. according to $\mathcal{D}$, the following inequalities hold:

$$\mathbb{E}[L(\mathcal{H}_{\mathrm{Rand}}(\mathbf{x}), \mathbf{y})] \leq \inf_{\mathsf{h} \in \mathsf{H}} \mathbb{E}[L(\mathsf{h}(\mathbf{x}), \mathbf{y})] + 2M\sqrt{\frac{l \log p}{T}} + 2M\sqrt{\frac{\log \frac{2}{\delta}}{T}}.$$

Learning Guarantees

Theorem

The following inequality relates the generalization error of the majority-vote algorithm to that of the randomized one:

$$\mathbb{E}[L_{\mathrm{Ham}}(\mathcal{H}_{\mathrm{MVote}}(\mathbf{x}), \mathbf{y})] \leq 2\ \mathbb{E}[L_{\mathrm{Ham}}(\mathcal{H}_{\mathrm{Rand}}(\mathbf{x}), \mathbf{y})]$$

where the expectations are taken over $(\mathbf{x}, \mathbf{y}) \sim \mathcal{D}$ *and* $\mathsf{h} \sim \mathsf{p}$.

Experiments

Table: Average Normalized Hamming Loss on synthetic data.

	ADS1, $m = 200$	ADS2, $m = 200$
$\mathcal{H}_{\text{MVote}}$	**0.0197 ± 0.00002**	**0.2172 ± 0.00983**
$\mathcal{H}_{\text{Boost}}$	**0.0197 ± 0.00002**	**0.2267 ± 0.00834**
$\mathcal{H}_{\text{SLE}}$	0.5641 ± 0.00044	0.2500 ± 0.05003
$\mathcal{H}_{\text{Rand}}$	0.1112 ± 0.00540	0.4000 ± 0.00018
BEST h_j	0.5635 ± 0.00004	0.4000

Table: Average Normalized Hamming Loss for ADS3.

$\mathcal{H}_{\text{MVote}}$	**0.1788 ± 0.00004**
$\mathcal{H}_{\text{Boost}}$	0.1831 ± 0.00240
$\mathcal{H}_{\text{SLE}}$	0.1954 ± 0.00185
$\mathcal{H}_{\text{Rand}}$	0.3196 ± 0.00018
Best h_j	0.2957 ± 0.00005

Experiments

Table: Average Normalized Hamming Loss, Penn Tree Bank.

	TR1, $m = 800$	TR2, $m = 1000$
$\mathcal{H}_{\text{MVote}}$	**0.0850 ± 0.00096**	**0.0746 ± 0.00014**
$\mathcal{H}_{\text{Boost}}$	0.1041 ± 0.00056	0.1414 ± 0.00233
$\mathcal{H}_{\text{SLE}}$	**0.0778 ± 0.00934**	**0.0814 ± 0.02558**
$\mathcal{H}_{\text{Rand}}$	0.1128 ± 0.00048	0.1652 ± 0.00077
BEST h_j	0.1032 ± 0.00007	0.1415 ± 0.00005

Table: Average Normalized Hamming Loss for OCR.

$\mathcal{H}_{\text{MVote}}$	0.1992 ± 0.00274
$\mathcal{H}_{\text{ESPBoost}}$	0.1992 ± 0.00274
$\mathcal{H}_{\text{SLE}}$	0.1994 ± 0.00307
$\mathcal{H}_{\text{Rand}}$	0.1994 ± 0.00276
Best h_j	0.1994 ± 0.00306

Experiments

Table: Average Normalized Hamming Loss, Pronunciation

	PDS1, $m = 130$	PDS2, $m = 400$
$\mathcal{H}_{\text{MVote}}$	**0.2225 ± 0.00301**	**0.2323 ±0.00069**
$\mathcal{H}_{\text{Boost}}$	0.3625 ± 0.01054	0.3499 ± 0.00509
$\mathcal{H}_{\text{SLE}}$	0.3130 ± 0.05137	0.3308 ± 0.03182
$\mathcal{H}_{\text{Rand}}$	0.4713 ± 0.00360	0.4607 ± 0.00131
BEST h_j	0.3449 ± 0.00368	0.3413 ± 0.00067

Table: Average edit-distance, Pronunciation

	PDS1, $m = 130$	PDS2, $m = 400$
$\mathcal{H}_{\text{MVote}}$	**0.8395 ± 0.01076**	**0.9626 ± 0.00341**
$\mathcal{H}_{\text{Boost}}$	1.3977 ± 0.06017	1.4092 ± 0.04352
$\mathcal{H}_{\text{SLE}}$	1.1762 ± 0.12530	1.2477 ± 0.12267
$\mathcal{H}_{\text{Rand}}$	1.8962 ± 0.01064	2.0838 ± 0.00518
BEST h_j	1.2163 ± 0.00619	1.2883 ± 0.00219

Experiments

Table: Average Normalized Hamming Loss, Speech

	$p = 5,\ m = 1500$	$p = 10,\ m = 1200$
$\mathcal{H}_{\text{MVote}}$	**0.2465 ± 0.00248**	**0.2606 ± 0.00320**
$\mathcal{H}_{\text{Boost}}$	0.2572 ± 0.00062	0.2864 ± 0.00103
$\mathcal{H}_{\text{SLE}}$	0.2572 ± 0.00061	0.2864 ± 0.00102
$\mathcal{H}_{\text{Rand}}$	0.2877 ± 0.00480	0.3430 ± 0.00468
BEST h_j	0.2573 ± 0.00060	0.2865 ± 0.00101

Table: Average Normalized Hamming Loss, Speech

	$p = 20,\ m = 900$	$p = 50,\ m = 700$
$\mathcal{H}_{\text{MVote}}$	**0.2773 ± 0.00139**	**0.3217 ± 0.00375**
$\mathcal{H}_{\text{Boost}}$	0.3115 ± 0.00089	0.3426 ± 0.00071
$\mathcal{H}_{\text{SLE}}$	0.3114 ± 0.00087	0.3425 ± 0.00076
$\mathcal{H}_{\text{Rand}}$	0.3977 ± 0.00302	0.4608 ± 0.00303
BEST h_j	0.3116 ± 0.00087	0.3427 ± 0.00077

Non-Additive Losses

- The natural loss functions for most of the key application with structured experts are **non-additive**:
  - machine translation (BLEU score).
  - speech recognition and natural language processing (edit-distance).
  - computational biology (n-gram similarity measures).
- But existing path experts algorithms cannot be applied with non-additive losses.

Solution for Non-Additive Losses

- Two new broad families of loss functions: Rational and Tropical Losses.
- Extensions of FPL and RWM to these loss functions based on powerful weighted automata and transducers algorithms.

Conclusions

- Ensemble methods for structured prediction with learning guarantees.
- On-line and Boosting algorithms.
- Good performance on real and synthetic data.
- Extensions to non-additive losses (e.g. edit-distance).