Learning Structured Predictors from Bandit Feedback for Interactive NLP

Artem Sokolov[◇,*], **Julia Kreutzer**[*], **Christopher Lo**[†,*], **Stefan Riezler**[‡,*]

[*]Computational Linguistics & [‡]IWR, Heidelberg University, Germany
[†]Dept. of Mathematics, Tufts University, USA

[◇]Amazon Development Center, Germany

Example: Learning SMT from Human Post-Edits

Example: Learning SMT from Human Post-Edits

- Data:
  - ➡ cost of professional translators
  - ➡ required editor expertise
  - ➡ slow in general

Example: Learning SMT from Human Post-Edits

- Data:
  - ➡ cost of professional translators
  - ➡ required editor expertise
  - ➡ slow in general
- Learning:
  - ➡ unclear mapping of post-edits to SMT operations, reachability
  - ➡ editors omit/add information, rewrite from scratch
  - ➡ small total number of post-edits

Example: Learning SMT from Human Post-Edits

- Data:
  - ➡ cost of professional translators
  - ➡ required editor expertise
  - ➡ slow in general
- Learning:
  - ➡ unclear mapping of post-edits to SMT operations, reachability
  - ➡ editors omit/add information, rewrite from scratch
  - ➡ small total number of post-edits
- Resulting model:
  - ➡ mismatch between human editors and real users

Example: Learning SMT from Human Post-Edits

- Data:
  - ➡ cost of professional translators
  - ➡ required editor expertise
  - ➡ slow in general
- Learning:
  - ➡ unclear mapping of post-edits to SMT operations, reachability
  - ➡ editors omit/add information, rewrite from scratch
  - ➡ small total number of post-edits
- Resulting model:
  - ➡ mismatch between human editors and real users

Ideally we need

- weaker-than-post-edit feedbacks
- that are easy to directly elicit from users
- fast learning

Online Bandit Learning

1. observe input structure x_t
2. propose output structure y_t
3. receive feedback to y_t (e.g. task loss, but not the true y)
4. update parameters

Online Bandit Learning

1. observe input structure x_t
2. propose output structure y_t
3. receive feedback to y_t (e.g. task loss, but not the true y)
4. update parameters

Learner does not know correct structure nor what would have happened if it had predicted differently

How to Learn from User Feedback?

Online Bandit Learning

1. observe input structure x_t
2. propose output structure y_t
3. receive feedback to y_t (e.g. task loss, but not the true y)
4. update parameters

Learner does not know correct structure nor what would have happened if it had predicted differently

‘One-armed bandits’ (slot machines)

- have to find a machine that gives you most money
- can try only one machine per time
- exploration/exploitation dilemma

- *learning from bandit feedback*
  - ➡ goal: minimize expected regret for selecting an arm
  - ➡ set of arms is usually small Auer et al. (2002b,a)
  - ➡ this work: exponential set of arms (outputs)
  - ➡ stochastic assumptions on the input but not on the feedback + context

- *learning from bandit feedback*
  - ➡ goal: minimize expected regret for selecting an arm
  - ➡ set of arms is usually small Auer et al. (2002b,a)
  - ➡ this work: exponential set of arms (outputs)
  - ➡ stochastic assumptions on the input but not on the feedback + context
- *reinforcement learning*
  - ➡ goal: maximize expected reward in an MDP
  - ➡ closest approach: policy gradient Sutton et al. (2000)
  - ➡ this work can be seen as one-state MDP
  - ➡ action = structured output

- *learning from bandit feedback*
  - ➡ goal: minimize expected regret for selecting an arm
  - ➡ set of arms is usually small Auer et al. (2002b,a)
  - ➡ this work: exponential set of arms (outputs)
  - ➡ stochastic assumptions on the input but not on the feedback + context
- *reinforcement learning*
  - ➡ goal: maximize expected reward in an MDP
  - ➡ closest approach: policy gradient Sutton et al. (2000)
  - ➡ this work can be seen as one-state MDP
  - ➡ action = structured output
- *pairwise preference* learning
  - ➡ full information setting
  - ➡ analysized under zero order optimization Yue and Joachims (2009); Agarwal et al. (2010)
  - ➡ this work: stochastic first-order optimization approach

Many potential NLP applications:

- numerical judgments on output quality
  - ➡ action learning Branavan et al. (2009)
  - ➡ machine translation Sokolov et al. (2015)
    - requires impractically many feedback
    - numerical feedback is hard to elicit

Many potential NLP applications:

- numerical judgments on output quality
  - ➡ action learning Branavan et al. (2009)
  - ➡ machine translation Sokolov et al. (2015)
    - requires impractically many feedback
    - numerical feedback is hard to elicit

This Work

- extending previous work with focus on
  1. learning speed: by strong convexification of the objective
  2. elicitability: by learning from pairwise preferences
- ‘banditize’ two new objectives
- empirical evaluation on several NLP tasks

- underlying Gibbs distribution

$$p_w(y|x) \propto e^{w^\top \phi(x,y)}$$

- $\Delta_y(y'; x)$ – loss for predicting y' instead of y
- expected loss (*aka* risk) Och (2003); Gimpel and Smith (2010); Yuille and He (2012)

$$J(w) = \mathbb{E}_{p(x,y)p_w(y'|x)}\left[\Delta_y(y')\right]$$

- underlying Gibbs distribution

$$p_w(y|x) \propto e^{w^\top \phi(x,y)}$$

- $\Delta_y(y'; x)$ – loss for predicting y' instead of y
- expected loss (*aka* risk) Och (2003); Gimpel and Smith (2010); Yuille and He (2012)

$$J(w) = \mathbb{E}_{p(x,y)p_w(y'|x)} \left[\Delta_y(y')\right]$$

Full Information

- expected loss is replaced by empirical risk minimization

$$J(w) = \frac{1}{T}\sum_{t=0}^{T} \mathbb{E}_{p_w(y'|x_t)} \Delta_{y_t}(y') p_w(y'|x_t)$$

- continuous and differentiable, although typically non-convex
- most approaches rely on gradient techniques
- need to know gold-standard y_t to calculate $\Delta_{y_t}(y')$ and
- evaluate it for all y' in the expectation

- what to do if the gold-standard y_t is unknown and
- we cannot evaluate all candidates y'?

- what to do if the gold-standard y_t is unknown and
- we cannot evaluate all candidates y'?
- pass the evaluation of $\Delta(y')$ to the user (dropping y_t in the subscript)
- replace gradient with its unbiased estimate

- what to do if the gold-standard y_t is unknown and
- we cannot evaluate all candidates y'?
- pass the evaluation of $\Delta(y')$ to the user (dropping y_t in the subscript)
- replace gradient with its unbiased estimate

Learning with Bandit Information

1: Input: learning rate γ
2: Initialize w_0
3: **for** $t = 0, \ldots, T$ **do**
4: Observe x_t
5: Sample $\tilde{y}_t \sim p_{w_t}(y|x_t)$
6: Obtain feedback $\Delta(\tilde{y}_t)$
7: Update $w_{t+1} = w_t - \gamma\ s_t$
8: Choose a solution $\hat{w}$ from the list $\{w_0, \ldots, w_T\}$

- what to do if the gold-standard y_t is unknown and
- we cannot evaluate all candidates y'?
- pass the evaluation of $\Delta(y')$ to the user (dropping y_t in the subscript)
- replace gradient with its unbiased estimate

Learning with Bandit Information

1: Input: learning rate γ
2: Initialize w_0
3: **for** $t = 0, \ldots, T$ **do**
4: Observe x_t
5: Sample $\tilde{y}_t \sim p_{w_t}(y|x_t)$ simultaneous exploration/exploitation
6: Obtain feedback $\Delta(\tilde{y}_t)$
7: Update $w_{t+1} = w_t - \gamma\, s_t$
8: Choose a solution $\hat{w}$ from the list $\{w_0, \ldots, w_T\}$

Bandit Information

- what to do if the gold-standard y_t is unknown and
- we cannot evaluate all candidates y'?
- pass the evaluation of $\Delta(y')$ to the user (dropping y_t in the subscript)
- replace gradient with its unbiased estimate

Learning with Bandit Information

```
Input: learning rate γ
Initialize w_0
for t = 0, ..., T do
    Observe x_t
    Sample ỹ_t ~ p_{w_t}(y|x_t)        simultaneous exploration/exploitation
    Obtain feedback Δ(ỹ_t)
    Update w_{t+1} = w_t − γ s_t        E_x E_ỹ[s_t] = ∇_w J
Choose a solution ŵ from the list {w_0, ..., w_T}
```

- Input: learning rate γ
- Initialize w_0
- **for** $t = 0, \dots, T$ **do**
  - Observe x_t
  - Sample $\tilde{y}_t \sim p_{w_t}(y|x_t)$ — simultaneous exploration/exploitation
  - Obtain feedback $\Delta(\tilde{y}_t)$
  - Update $w_{t+1} = w_t - \gamma\, s_t$ — $\mathbb{E}_x \mathbb{E}_{\tilde{y}}[s_t] = \nabla_w J$
- Choose a solution $\hat{w}$ from the list $\{w_0, \dots, w_T\}$

Instantiation for the expected loss Branavan et al. (2009); Sokolov et al. (2015)

$$J(w) = \mathbb{E}_x \mathbb{E}_y [\Delta(y)]$$
$$\tilde{y} \sim p_w(y|x)$$
$$s_t = \Delta(\tilde{y})\big(\phi(x, \tilde{y}) - \mathbb{E}_y[\phi(x, y)]\big)$$

Instantiation for the expected loss Branavan et al. (2009); Sokolov et al. (2015)

$$J(w) = \mathbb{E}_x \mathbb{E}_y[\Delta(y)]$$
$$\tilde{y} \sim p_w(y|x)$$
$$s_t = \Delta(\tilde{y})\big(\phi(x,\tilde{y}) - \mathbb{E}_y[\phi(x,y)]\big)$$

- non-convex stochastic first-order optimization
- converges to a local minimum Polyak and Tsypkin (1973)
- iteration complexity is $\mathcal{O}(\varepsilon^{-2})$ Ghadimi and Lan (2012)
  i.e. number of steps until $\mathbb{E}[\|\nabla J(w_t)\|^2] \leq \varepsilon$

Instantiation for the expected loss Branavan et al. (2009); Sokolov et al. (2015)

$$J(w) = \mathbb{E}_x \mathbb{E}_y[\Delta(y)]$$
$$\tilde{y} \sim p_w(y|x)$$
$$s_t = \Delta(\tilde{y})\big(\phi(x,\tilde{y}) - \mathbb{E}_y[\phi(x,y)]\big)$$

- non-convex stochastic first-order optimization
- converges to a local minimum Polyak and Tsypkin (1973)
- iteration complexity is $\mathcal{O}(\varepsilon^{-2})$ Ghadimi and Lan (2012)
  i.e. number of steps until $\mathbb{E}[\|\nabla J(w_t)\|^2] \leq \varepsilon$

1. for easier feedback elicitability:
   - **pairwise preference loss**
2. for faster convergence: (strongly) convexify the loss to get $\mathcal{O}(\varepsilon^{-1})$ complexity
   - **cross-entropy loss**

1 Pairwise Loss

$$J(w) = \mathbb{E}_x \mathbb{E}_{\langle y_i, y_j \rangle}[\Delta(\langle y_i, y_j \rangle)]$$

$$\langle \tilde{y}_i, \tilde{y}_j \rangle \sim p_w(\langle y_i, y_j \rangle | x) \propto e^{w^\top(\phi(x,y_i) - \phi(x,y_j))}$$

$$s_t = \Delta(\langle \tilde{y}_i, \tilde{y}_j \rangle)\big(\phi(x, \langle \tilde{y}_i, \tilde{y}_j \rangle) - \mathbb{E}_{\langle y_i, y_j \rangle}[\phi(x, \langle y_i, y_j \rangle)]\big)$$

1 Pairwise Loss

$$J(w) = \mathbb{E}_x \mathbb{E}_{\langle y_i, y_j \rangle}[\Delta(\langle y_i, y_j \rangle)]$$

$$\langle \tilde{y}_i, \tilde{y}_j \rangle \sim p_w(\langle y_i, y_j \rangle | x) \propto e^{w^\top(\phi(x,y_i) - \phi(x,y_j))}$$

$$s_t = \Delta(\langle \tilde{y}_i, \tilde{y}_j \rangle)\big(\phi(x, \langle \tilde{y}_i, \tilde{y}_j \rangle) - \mathbb{E}_{\langle y_i, y_j \rangle}[\phi(x, \langle y_i, y_j \rangle)]\big)$$

- ➡ arguably easier for users to judge (binary judgment) Thurstone (1927)
- ➡ but it's just expected loss on pairs, so still $\mathcal{O}(\varepsilon^{-2})$ complexity

1 Pairwise Loss

$$J(w) = \mathbb{E}_x \mathbb{E}_{\langle y_i, y_j \rangle}[\Delta(\langle y_i, y_j \rangle)]$$

$$\langle \tilde{y}_i, \tilde{y}_j \rangle \sim p_w(\langle y_i, y_j \rangle | x) \propto e^{w^\top (\phi(x,y_i) - \phi(x,y_j))}$$

$$s_t = \Delta(\langle \tilde{y}_i, \tilde{y}_j \rangle) \big(\phi(x, \langle \tilde{y}_i, \tilde{y}_j \rangle) - \mathbb{E}_{\langle y_i, y_j \rangle}[\phi(x, \langle y_i, y_j \rangle)]\big)$$

- ➡ arguably easier for users to judge (binary judgment) Thurstone (1927)
- ➡ but it's just expected loss on pairs, so still $\mathcal{O}(\varepsilon^{-2})$ complexity

2 Cross-Entropy

$$J(w) = \mathbb{E}_x \mathbb{E}_{g(y)}[-\log p_w(y|x)], \text{ gain function } g(y) = 1 - \Delta(y)$$

$$\tilde{y} \sim p_w(y|x)$$

$$s_t = \frac{1 - \Delta(\tilde{y})}{p_w(\tilde{y}|x)} \big(- \phi(x, \tilde{y}) + \mathbb{E}_y[\phi(x, y)]\big)$$

1 Pairwise Loss

$$J(w) = \mathbb{E}_x \mathbb{E}_{\langle y_i, y_j \rangle}[\Delta(\langle y_i, y_j \rangle)]$$

$$\langle \tilde{y}_i, \tilde{y}_j \rangle \sim p_w(\langle y_i, y_j \rangle | x) \propto e^{w^\top(\phi(x,y_i) - \phi(x,y_j))}$$

$$s_t = \Delta(\langle \tilde{y}_i, \tilde{y}_j \rangle)\big(\phi(x, \langle \tilde{y}_i, \tilde{y}_j \rangle) - \mathbb{E}_{\langle y_i, y_j \rangle}[\phi(x, \langle y_i, y_j \rangle)]\big)$$

- ➡ arguably easier for users to judge (binary judgment) Thurstone (1927)
- ➡ but it's just expected loss on pairs, so still $\mathcal{O}(\varepsilon^{-2})$ complexity

2 Cross-Entropy

$$J(w) = \mathbb{E}_x \mathbb{E}_{g(y)}[-\log p_w(y|x)], \text{ gain function } g(y) = 1 - \Delta(y)$$

$$\tilde{y} \sim p_w(y|x)$$

$$s_t = \frac{1 - \Delta(\tilde{y})}{p_w(\tilde{y}|x)}\big(- \phi(x, \tilde{y}) + \mathbb{E}_y[\phi(x, y)]\big)$$

- ➡ can be made strongly convex by adding a regularizer
- ➡ expecting faster $\mathcal{O}(\varepsilon^{-1})$ convergence
- ➡ this loss upper bounds the expected loss, if $g(y)$ is a distribution
- ➡ but in the bandit setup normalizing is not possible

task	features	structure	task loss Δ	dataset
text class.	sparse	4 classes	error rate	RCV1
word OCR	dense	CRF	Hamming	Taskar et al. (2003)
NP-chunking	sparse	bigram-CRF	F1	CoNLL-2000
SMT	dense sparse	n-best list hypergraph	BLEU	EuroParl$\rightarrow$ NewsComm

task	features	structure	task loss Δ	dataset
text class.	sparse	4 classes	error rate	RCV1
word OCR	dense	CRF	Hamming	Taskar et al. (2003)
NP-chunking	sparse	bigram-CRF	F1	CoNLL-2000
SMT	dense sparse	n-best list hypergraph	BLEU	EuroParl$\to$ NewsComm

Setup

- simulated bandit feedback by evaluating task loss against gold-standard structures *without* revealing them to the learner
- constant learning rates in most experiments, ℓ_2-regularization, momentum, annealing
- empirical convergence assessed as the $\#$ of steps before overfitting on dev
- test results for the best model found on dev (under MAP inference, averaged)

Results

	task	loss/gain	full information		partial information: expected loss	pairwise	cross-entropy
	Text classification	0/1 ↓	percep., $\lambda = 10^{-6}$	0.040	0.031	0.083	0.035
CRF	Word OCR (dense)	Hamming ↓	likelihood	0.099	0.261	0.332	0.257
CRF	Chunking (sparse)	F1-score ↑	likelihood	0.935	0.923	0.914	0.891
			out-of-domain	**in-domain**			
SMT	News (n-best list, dense)	BLEU ↑	0.259	0.284	0.269	0.275	0.276
SMT	News (hypergraph, sparse)	BLEU ↑	0.265	0.283	0.267	0.273	0.271

Results

	task	loss/gain	full information		partial information expected loss	pairwise	cross-entropy
	Text classification	0/1 ↓	percep., $\lambda = 10^{-6}$	0.040	0.031	0.083	0.035
CRF	Word OCR (dense)	Hamming ↓	likelihood	0.099	0.261	0.332	0.257
CRF	Chunking (sparse)	F1-score ↑	likelihood	0.935	0.923	0.914	0.891
			out-of-domain	**in-domain**			
SMT	News (n-best list, dense)	BLEU ↑	0.259	0.284	0.269	0.275	0.276
SMT	News (hypergraph, sparse)	BLEU ↑	0.265	0.283	0.267	0.273	0.271

Results

	task	loss/gain	full information		partial information expected loss	pairwise	cross-entropy
	Text classification	0/1 ↓	percep., $\lambda = 10^{-6}$	0.040	0.031	0.083	0.035
CRF	**Word OCR (dense)**	Hamming ↓	likelihood	0.099	0.261	0.332	0.257
CRF	Chunking (sparse)	F1-score ↑	likelihood	0.935	0.923	0.914	0.891
			out-of-domain	**in-domain**			
SMT	News (n-best list, dense)	BLEU ↑	0.259	0.284	0.269	0.275	0.276
SMT	News (hypergraph, sparse)	BLEU ↑	0.265	0.283	0.267	0.273	0.271

Results

	task	loss/gain	full information		partial information expected loss	pairwise	cross-entropy
	Text classification	0/1 ↓	percep., $\lambda = 10^{-6}$	0.040	0.031	0.083	0.035
CRF	Word OCR (dense)	Hamming ↓	likelihood	0.099	0.261	0.332	0.257
CRF	**Chunking (sparse)**	F1-score ↑	likelihood	0.935	0.923	0.914	0.891
			out-of-domain	**in-domain**			
SMT	News (n-best list, dense)	BLEU ↑	0.259	0.284	0.269	0.275	0.276
SMT	News (hypergraph, sparse)	BLEU ↑	0.265	0.283	0.267	0.273	0.271

Results

	task	loss/gain	full information		partial information: expected loss	partial information: pairwise	partial information: cross-entropy
	Text classification	0/1 ↓	percep., $\lambda = 10^{-6}$	0.040	0.031	0.083	0.035
CRF	Word OCR (dense)	Hamming ↓	likelihood	0.099	0.261	0.332	0.257
CRF	Chunking (sparse)	F1-score ↑	likelihood	0.935	0.923	0.914	0.891
			out-of-domain	**in-domain**			
SMT	**News (n-best list, dense)**	BLEU ↑	0.259	0.284	0.269	0.275	0.276
SMT	**News (hypergraph, sparse)**	BLEU ↑	0.265	0.283	0.267	0.273	0.271

Results

	task	loss/gain	full information		partial information: expected loss	pairwise	cross-entropy
	Text classification	0/1 ↓	percep., $\lambda = 10^{-6}$	0.040	0.031	0.083	0.035
CRF	Word OCR (dense)	Hamming ↓	likelihood	0.099	0.261	0.332	0.257
CRF	Chunking (sparse)	F1-score ↑	likelihood	0.935	0.923	0.914	0.891
			out-of-domain	**in-domain**			
SMT	News (n-best list, dense)	BLEU ↑	0.259	0.284	0.269	0.275	0.276
SMT	News (hypergraph, sparse)		0.265	0.283	0.267	0.273	0.271

Iterations to meet stopping criterion on dev data

	theory	$\mathcal{O}(\varepsilon^{-2})$	$\mathcal{O}(\varepsilon^{-2})$	$\mathcal{O}(\varepsilon^{-1})$
	task\loss	**expected loss**	**pairwise**	**cross-entropy**
	Text classification	2.0M	**0.5M**	1.1M
CRF	Word OCR	14.4M	**9.3M**	37.9M
CRF	Chunking	7.5M	**4.7M**	5.9M
SMT	News (n-best, dense)	3.8M	**1.2M**	1.2M
SMT	News (h-graph, sparse)	370k	**115k**	281k

Possible reasons

- different hidden constants in the $\mathcal{O}(\cdot)$ notations
- in particular, high variance σ^2

$$\mathbb{E}[\|\nabla J(w_T)\|^2] \propto \frac{L^2}{T} + \mathsf{const} \cdot \frac{L\sigma}{\sqrt{T}}$$ Ghadimi and Lan (2012)

Possible reasons

- different hidden constants in the $\mathcal{O}(\cdot)$ notations
- in particular, high variance σ^2

$$\mathbb{E}[\|\nabla J(w_T)\|^2] \propto \frac{L^2}{T} + \text{const} \cdot \frac{L\sigma}{\sqrt{T}}$$ Ghadimi and Lan (2012)

We empirically estimated (same T and γ, SMT hypergraph task):

- average gradient norm $\langle \|s_T\|^2 \rangle$
- Lipschitz constant L of the gradient ∇J as $\max_{t,t'} \frac{\|s_t - s_{t'}\|}{\|w_t - w_{t'}\|}$
- variance σ^2 as $\max_{t=0,\dots T} \|s_t - \frac{1}{T}\sum_{t=0}^{T} s_t\|^2$

	$\langle \|s_T\|^2 \rangle$	L	σ^2
expected loss	$0.02_{\pm 0.03}$	$11_{\pm 12}$	$0.7_{\pm 0.9}$
pairwise	$2e\text{-}6_{\pm 3e-8}$	**$0.08_{\pm 0.01}$**	**$0.0008_{\pm 0.0000}$**
cross-entropy	$3.04_{\pm 0.02}$	$0.62_{\pm 0.2}$	$677_{\pm 115}$

Possible reasons

- different hidden constants in the $\mathcal{O}(\cdot)$ notations
- in particular, high variance σ^2

$$\mathbb{E}[\|\nabla J(w_T)\|^2] \propto \frac{L^2}{T} + \text{const} \cdot \frac{L\sigma}{\sqrt{T}}$$ Ghadimi and Lan (2012)

We empirically estimated (same T and γ, SMT hypergraph task):

- average gradient norm $\langle\|s_T\|^2\rangle$
- Lipschitz constant L of the gradient ∇J as $\max_{t,t'} \frac{\|s_t - s_{t'}\|}{\|w_t - w_{t'}\|}$
- variance σ^2 as $\max_{t=0,\dots T} \|s_t - \frac{1}{T}\sum_{t=0}^{T} s_t\|^2$

	$\langle\|s_T\|^2\rangle$	L	σ^2
expected loss	$0.02_{\pm 0.03}$	$11_{\pm 12}$	$0.7_{\pm 0.9}$
pairwise	2e-6$_{\pm 3e-8}$	$0.08_{\pm 0.01}$	$0.0008_{\pm 0.0000}$
cross-entropy	$3.04_{\pm 0.02}$	$0.62_{\pm 0.2}$	**$677_{\pm 115}$**

- **two new objectives** for learning structured predictors from weak feeedback
  - ➡ applicable to cases with no gold-standard structures and only feedback available
- consistent **advantage of pairwise feedback**
  - ➡ surprising, since theory predicts the fastest convergence for strongly convex losses
  - ➡ can be explained by empirical factors: variance, Lipschitz constant
- additionally, pairwise learning requires only **relative feedback** (good for users)

- **two new objectives** for learning structured predictors from weak feeedback
  - ➡ applicable to cases with no gold-standard structures and only feedback available
- consistent **advantage of pairwise feedback**
  - ➡ surprising, since theory predicts the fastest convergence for strongly convex losses
  - ➡ can be explained by empirical factors: variance, Lipschitz constant
- additionally, pairwise learning requires only **relative feedback** (good for users)

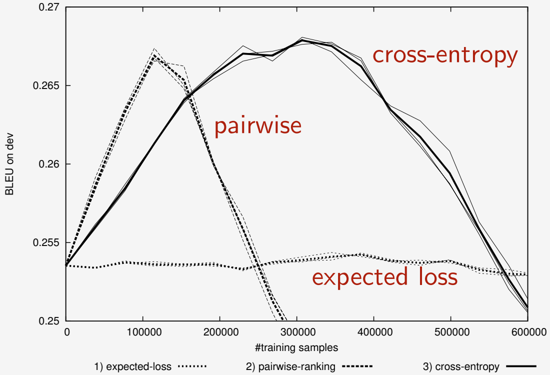
Thank you!

Acknowledgements:

DFG Deutsche Forschungsgemeinschaft

amazon

- SMT hypergraph re-decoding on the development set
- averaged over 3 independent runs

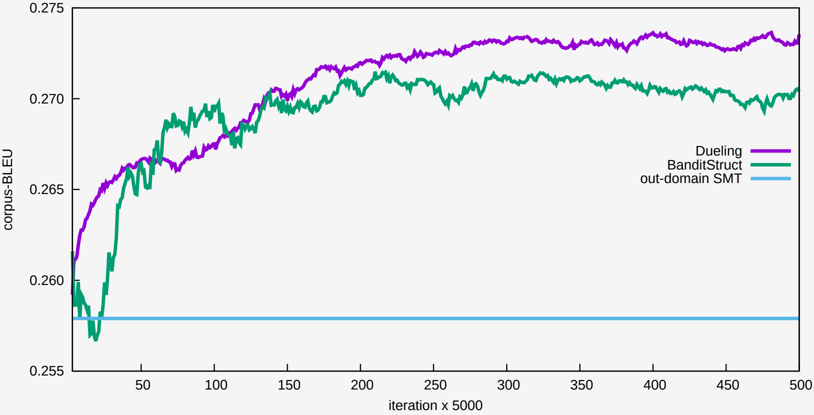


- pairwise ranking reaches peak performance fastest
- still large variance of cross-entropy learning (despite clipping)

	task	expected loss	pairwise	cross-entropy
	Text classification	$\gamma_t = 1.0$	$\gamma_t = 10^{-0.75}$	$\gamma_t = 10^{-1}$
CRF	OCR	$T_0 = 0.4, \gamma_t = 10^{-3.5}$	$T_0 = 0.1, \gamma_t = 10^{-4}$	$\lambda = 10^{-5}, k = 10^{-2}, \gamma_t = 10^{-6}$
CRF	Chunking	$\gamma_t = 10^{-4}$	$\gamma_t = 10^{-4}$	$\lambda = 10^{-6}, k = 10^{-2}, \gamma_t = 10^{-6}$
SMT	News (n-best, dense)	$\gamma_t = 10^{-5}$	$\gamma_t = 10^{-4.75}$	$\lambda = 10^{-4}, \mu = 0.99, \gamma_t = 10^{-6}/\sqrt{t}$
SMT	News (h-graph, sparse)	$\gamma_t = 10^{-5}$	$\gamma_t = 10^{-4}$	$\lambda = 10^{-6}, k = 5 \cdot 10^{-3}, \gamma_t = 10^{-6}$

Table: Metaparameter settings determined on *dev* sets for constant learning rate γ_t, temperature coefficient T_0 for annealing under the schedule $T = T_0/\sqrt[3]{\mathrm{epoch}+1}$, momentum coefficient $\min\{1 - 1/(t/2+2), \mu\}$, clipping constant k used to replace $p_{w_t}(\tilde{y}_t|x_t)$ with $\max\{p_{w_t}(\tilde{y}_t|x_t), k\}$, ℓ_2 regularization constant λ. Unspecified parameters are set to zero.

full information		bandit information	
in-domain SMT	**out-domain SMT**	**dueling bandits**	**expected loss**
0.2854	0.2579	$0.2731_{\pm 0.001}$	$0.2705_{\pm 0.001}$



Agarwal, A., Dekel, O., and Xiao, L. (2010). Optimal algorithms for online convex optimization with multi-point bandit feedback. In *COLT*, Haifa, Israel.

Auer, P., Cesa-Bianchi, N., and Fischer, P. (2002a). Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47:235–256.

Auer, P., Cesa-Bianchi, N., Freund, Y., and Schapire, R. E. (2002b). The nonstochastic multiarmed bandit problem. *SIAM J. on Computing*, 32(1):48–77.

Branavan, S., Chen, H., Zettlemoyer, L. S., and Barzilay, R. (2009). Reinforcement learning for mapping instructions to actions. In *ACL*, Suntec, Singapore.

Ghadimi, S. and Lan, G. (2012). Stochastic first- and zeroth-order methods for nonconvex stochastic programming. *SIAM J. on Optimization*, 4(23):2342–2368.

Gimpel, K. and Smith, N. A. (2010). Softmax-margin training for structured log-linear models. Technical Report CMU-LTI-10-008, Carnegie Mellon University, Pittsburgh, PA.

Och, F. J. (2003). Minimum error rate training in statistical machine translation. In *HLT-NAACL*, Edmonton, Canada.

Polyak, B. T. and Tsypkin, Y. Z. (1973). Pseudogradient adaptation and training algorithms. *Automation and remote control*, 34(3):377–397.

Sokolov, A., Riezler, S., and Urvoy, T. (2015). Bandit structured prediction for learning from user feedback in statistical machine translation. In *MT Summit XV*, Miami, FL.

Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. (2000). Policy gradient methods for reinforcement learning with function approximation. In *NIPS*, Vancouver, Canada.

Taskar, B., Guestrin, C., and Koller, D. (2003). Max-margin markov networks. In *NIPS*, Vancouver, Canada.

Thurstone, L. L. (1927). A law of comparative judgement. *Psychological Review*, 34:278–286.

Yue, Y. and Joachims, T. (2009). Interactively optimizing information retrieval systems as a dueling bandits problem. In *ICML*, Montreal, Canada.

Yuille, A. and He, X. (2012). Probabilistic models of vision and max-margin methods. *Frontiers of Electrical and Electronic Engineering*, 7(1):94–106.