Multi-Task Learning from Large-Scale High-Dimensional Data

Stefan Riezler*
(joint work with Patrick Simianer* and Chris Dyer†)

* Department of Computational Linguistics, Heidelberg University, Germany
† Language Technologies Institute, Carnegie Mellon University, Pittsburgh, PA

Big Data

- **Data** can be characterized as **big** by
  - large size of training set,
  - high dimensionality of feature representation of data.
- Not all datasets advertised as “large” meet both requirements (e.g. Learning-to-Rank Challenges at Yahoo! and Microsoft work on *hundreds* of features for *tens of thousands* of queries)
- Our application scenario is **Statistical Machine Translation (SMT)**, using *billions* of features and training examples.

Big Data

- **Data** can be characterized as **big** by
  - large size of training set,
  - high dimensionality of feature representation of data.
- Not all datasets advertised as “large” meet both requirements (e.g. Learning-to-Rank Challenges at Yahoo! and Microsoft work on *hundreds* of features for *tens of thousands* of queries)
- Our application scenario is **Statistical Machine Translation (SMT)**, using *billions* of features and training examples.

Big Data

- **Data** can be characterized as **big** by
  - large size of training set,
  - high dimensionality of feature representation of data.
- Not all datasets advertised as "large" meet both requirements (e.g. Learning-to-Rank Challenges at Yahoo! and Microsoft work on *hundreds* of features for *tens of thousands* of queries)
- Our application scenario is **Statistical Machine Translation (SMT)**, using *billions* of features and training examples.

Large Scale Learning

- **Learning problem** is **large scale** if
  - training data cannot be stored in RAM (Langford on http://hunch.net/?p=330, 2008),
  - time constraint requires that algorithms scale at worst linearly with number of examples (Bottou & Bousquet NIPS'07).
- Solutions:
  - **Online learning** for linear scaling in training sample size (Bottou & Le Cun NIPS'04),
  - combined with **feature selection** for memory efficient feature representation (Langford et al. JMLR'09),
  - combined with **parallelization** and **averaging** for parallel acceleration and reduced variance at asymptotic online learning guarantees (Zinkevich et al. NIPS'10) .
- We add another dimension: **Multi-task learning**.

Large Scale Learning

- **Learning problem** is **large scale** if
  - training data cannot be stored in RAM (Langford on http://hunch.net/?p=330, 2008),
  - time constraint requires that algorithms scale at worst linearly with number of examples (Bottou & Bousquet NIPS'07).
- Solutions:
  - **Online learning** for linear scaling in training sample size (Bottou & Le Cun NIPS'04),
  - combined with **feature selection** for memory efficient feature representation (Langford et al. JMLR'09),
  - combined with **parallelization** and **averaging** for parallel acceleration and reduced variance at asymptotic online learning guarantees (Zinkevich et al. NIPS'10) .
- We add another dimension: **Multi-task learning**.

Large Scale Learning

- **Learning problem** is **large scale** if
  - training data cannot be stored in RAM (Langford on http://hunch.net/?p=330, 2008),
  - time constraint requires that algorithms scale at worst linearly with number of examples (Bottou & Bousquet NIPS'07).
- Solutions:
  - **Online learning** for linear scaling in training sample size (Bottou & Le Cun NIPS'04),
  - combined with **feature selection** for memory efficient feature representation (Langford et al. JMLR'09),
  - combined with **parallelization** and **averaging** for parallel acceleration and reduced variance at asymptotic online learning guarantees (Zinkevich et al. NIPS'10) .
- We add another dimension: **Multi-task learning**.

Multi-Task Learning

- **Goal:** A number of statistical models need to be estimated simultaneously from data belonging to different tasks.
- **Examples**:
  - OCR of handwritten characters from different writers: Exploit commonalities on pixel- or stroke-level shared between writers.
  - LTR from search engine query logs from different countries: Some queries are country-specific (“football”), most preference rankings are shared across countries.
- **Idea:**
  - Learn a shared model that takes advantage of commonalities among tasks, without neglecting individual knowledge.

Multi-Task Learning

- **Goal:** A number of statistical models need to be estimated simultaneously from data belonging to different tasks.
- **Examples**:
  - OCR of handwritten characters from different writers: Exploit commonalities on pixel- or stroke-level shared between writers.
  - LTR from search engine query logs from different countries: Some queries are country-specific (“football”), most preference rankings are shared across countries.
- **Idea:**
  - Learn a shared model that takes advantage of commonalities among tasks, without neglecting individual knowledge.

Multi-Task Learning

- **Goal:** A number of statistical models need to be estimated simultaneously from data belonging to different tasks.
- **Examples**:
  - OCR of handwritten characters from different writers: Exploit commonalities on pixel- or stroke-level shared between writers.
  - LTR from search engine query logs from different countries: Some queries are country-specific ("football"), most preference rankings are shared across countries.
- **Idea:**
  - Learn a shared model that takes advantage of commonalities among tasks, without neglecting individual knowledge.

Our Application: Learning from Big Data in SMT

- Machine learning theory and practice suggests benefits from using **expressive feature representations** and from **tuning on large training samples**.
- Discriminative training in SMT has mostly been content with tuning **small sets of dense features** on **small development data** (Och NAACL'03).
- Notable exceptions using **larger feature and training sets**:
  - Liang et al. ACL'06: 1.5M features, 67K parallel sentences.
  - Tillmann and Zhang ACL'06: 35M feats, 230K sents.
  - Blunsom et al. ACL'08: 7.8M feats, 100K sents.
  - Simianer, Riezler, Dyer ACL'12: 4.7M feats, 1.6M sents.
  - Flanigan, Dyer, Carbonell NAACL'13: 28.8M feats, 1M sents.

Our Application: Learning from Big Data in SMT

- Machine learning theory and practice suggests benefits from using **expressive feature representations** and from **tuning on large training samples**.
- Discriminative training in SMT has mostly been content with tuning **small sets of dense features** on **small development data** (Och NAACL'03).
- Notable exceptions using **larger feature and training sets**:
  - Liang et al. ACL'06: 1.5M features, 67K parallel sentences.
  - Tillmann and Zhang ACL'06: 35M feats, 230K sents.
  - Blunsom et al. ACL'08: 7.8M feats, 100K sents.
  - Simianer, Riezler, Dyer ACL'12: 4.7M feats, 1.6M sents.
  - Flanigan, Dyer, Carbonell NAACL'13: 28.8M feats, 1M sents.

Our Application: Learning from Big Data in SMT

- Machine learning theory and practice suggests benefits from using **expressive feature representations** and from **tuning on large training samples**.
- Discriminative training in SMT has mostly been content with tuning **small sets of dense features** on **small development data** (Och NAACL'03).
- Notable exceptions using **larger feature and training sets**:
  - Liang et al. ACL'06: 1.5M features, 67K parallel sentences.
  - Tillmann and Zhang ACL'06: 35M feats, 230K sents.
  - Blunsom et al. ACL'08: 7.8M feats, 100K sents.
  - Simianer, Riezler, Dyer ACL'12: 4.7M feats, 1.6M sents.
  - Flanigan, Dyer, Carbonell NAACL'13: 28.8M feats, 1M sents.

Our Approach: Multi-Task Distributed SGD

- **Distribute work and share information!**
  - **Online learning** via Stochastic Gradient Descent optimization.
  - **Distributed learning** using Hadoop/MapReduce or SunGridEngine.
  - **Feature selection** via ℓ_1/ℓ_2 block norm regularization on features across multiple tasks.
- **Pooling baseline:**
  - Concatenate data from all tasks into one big pool.
  - Becomes infeasible very quickly.
- **Independent modeling baseline :**
  - Independent training of task specific models.
  - Does not share any knowledge across tasks.

Our Approach: Multi-Task Distributed SGD

- **Distribute work and share information!**
  - **Online learning** via Stochastic Gradient Descent optimization.
  - **Distributed learning** using Hadoop/MapReduce or SunGridEngine.
  - **Feature selection** via ℓ_1/ℓ_2 block norm regularization on features across multiple tasks.
- **Pooling baseline:**
  - Concatenate data from all tasks into one big pool.
  - Becomes infeasible very quickly.
- **Independent modeling baseline :**
  - Independent training of task specific models.
  - Does not share any knowledge across tasks.

Our Approach: Multi-Task Distributed SGD

- **Distribute work and share information!**
  - **Online learning** via Stochastic Gradient Descent optimization.
  - **Distributed learning** using Hadoop/MapReduce or SunGridEngine.
  - **Feature selection** via ℓ_1/ℓ_2 block norm regularization on features across multiple tasks.
- **Pooling baseline:**
  - Concatenate data from all tasks into one big pool.
  - Becomes infeasible very quickly.
- **Independent modeling baseline :**
  - Independent training of task specific models.
  - Does not share any knowledge across tasks.

Our Approach: Multi-Task Distributed SGD

- **Distribute work and share information!**
  - **Online learning** via Stochastic Gradient Descent optimization.
  - **Distributed learning** using Hadoop/MapReduce or SunGridEngine.
  - **Feature selection** via ℓ_1/ℓ_2 block norm regularization on features across multiple tasks.
- **Pooling baseline:**
  - Concatenate data from all tasks into one big pool.
  - Becomes infeasible very quickly.
- **Independent modeling baseline :**
  - Independent training of task specific models.
  - Does not share any knowledge across tasks.

Our Approach: Multi-Task Distributed SGD

- **Distribute work and share information!**
  - **Online learning** via Stochastic Gradient Descent optimization.
  - **Distributed learning** using Hadoop/MapReduce or SunGridEngine.
  - **Feature selection** via ℓ_1/ℓ_2 block norm regularization on features across multiple tasks.
- **Pooling baseline:**
  - Concatenate data from all tasks into one big pool.
  - Becomes infeasible very quickly.
- **Independent modeling baseline :**
  - Independent training of task specific models.
  - Does not share any knowledge across tasks.

Our Approach: Multi-Task Distributed SGD

- **Distribute work and share information!**
  - **Online learning** via Stochastic Gradient Descent optimization.
  - **Distributed learning** using Hadoop/MapReduce or SunGridEngine.
  - **Feature selection** via ℓ_1/ℓ_2 block norm regularization on features across multiple tasks.
- **Pooling baseline:**
  - Concatenate data from all tasks into one big pool.
  - Becomes infeasible very quickly.
- **Independent modeling baseline :**
  - Independent training of task specific models.
  - Does not share any knowledge across tasks.

Related Work

- **Online learning**:
  - We deploy pairwise ranking perceptron (Shen & Joshi JMLR'05)
  - and margin perceptron (Collobert & Bengio ICML'04).
- **Distributed learning**:
  - Without feature selection, our algorithm reduces to Iterative Mixing (McDonald et al. NAACL'10),
  - which itself is related to Bagging (Breiman JMLR'96) if shards are treated as random samples.

Related Work

- **Online learning**:
  - We deploy pairwise ranking perceptron (Shen & Joshi JMLR'05)
  - and margin perceptron (Collobert & Bengio ICML'04).
- **Distributed learning**:
  - Without feature selection, our algorithm reduces to Iterative Mixing (McDonald et al. NAACL'10),
  - which itself is related to Bagging (Breiman JMLR'96) if shards are treated as random samples.

Related Work

- ℓ_1/ℓ_2 **regularization**:
  - Related to group-Lasso approaches which use mixed norms (Yuan & Lin JRSS'06), hierarchical norms (Zhao et al. Annals Stats'09), structured norms (Martins et al. EMNLP'11).
  - Difference: Norms and proximity operators are applied to *groups* of features in *single* regression or classification task – multi-task learning groups features orthogonally by tasks.
  - Closest relation to Obozinski et al. StatComput'10: Our algorithm is weight-based backward feature elimination variant of their gradient-based forward feature selection algorithm.

OL Framework: Pairwise Ranking Perceptron

- Preference pairs $\mathbf{x}_j = (\mathbf{x}_j^{(1)}, \mathbf{x}_j^{(2)})$ where $\mathbf{x}_j^{(1)}$ is ordered above $\mathbf{x}_j^{(2)}$ w.r.t. sentence-wise BLEU (Nakov et al. COLING'12).
- Hinge loss-type objective

$$l_j(\mathbf{w}) = (- \langle \mathbf{w}, \bar{\mathbf{x}}_j \rangle)_+$$

where $\bar{\mathbf{x}}_j = \mathbf{x}_j^{(1)} - \mathbf{x}_j^{(2)}$, $(a)_+ = \max(0, a)$, $\mathbf{w} \in \mathbb{R}^D$ is a weight vector, and $\langle \cdot, \cdot \rangle$ denotes the standard vector dot product.

- **Ranking perceptron** by stochastic subgradient descent:

$$\nabla l_j(\mathbf{w}) = \begin{cases} -\bar{\mathbf{x}}_j & \text{if } \langle \mathbf{w}, \bar{\mathbf{x}}_j \rangle \leq 0, \\ 0 & \text{else.} \end{cases}$$

OL Framework: Pairwise Ranking Perceptron

- Preference pairs $\mathbf{x}_j = (\mathbf{x}_j^{(1)}, \mathbf{x}_j^{(2)})$ where $\mathbf{x}_j^{(1)}$ is ordered above $\mathbf{x}_j^{(2)}$ w.r.t. sentence-wise BLEU (Nakov et al. COLING'12).
- Hinge loss-type objective

$$l_j(\mathbf{w}) = (-\langle \mathbf{w}, \bar{\mathbf{x}}_j \rangle)_+$$

where $\bar{\mathbf{x}}_j = \mathbf{x}_j^{(1)} - \mathbf{x}_j^{(2)}$, $(a)_+ = \max(0, a)$, $\mathbf{w} \in \mathbb{R}^D$ is a weight vector, and $\langle \cdot, \cdot \rangle$ denotes the standard vector dot product.

- **Ranking perceptron** by stochastic subgradient descent:

$$\nabla l_j(\mathbf{w}) = \begin{cases} -\bar{\mathbf{x}}_j & \text{if } \langle \mathbf{w}, \bar{\mathbf{x}}_j \rangle \leq 0, \\ 0 & \text{else.} \end{cases}$$

OL Framework: Pairwise Ranking Perceptron

- Preference pairs $\mathbf{x}_j = (\mathbf{x}_j^{(1)}, \mathbf{x}_j^{(2)})$ where $\mathbf{x}_j^{(1)}$ is ordered above $\mathbf{x}_j^{(2)}$ w.r.t. sentence-wise BLEU (Nakov et al. COLING'12).
- Hinge loss-type objective

$$l_j(\mathbf{w}) = (-\langle \mathbf{w}, \bar{\mathbf{x}}_j \rangle)_+$$

where $\bar{\mathbf{x}}_j = \mathbf{x}_j^{(1)} - \mathbf{x}_j^{(2)}$, $(a)_+ = \max(0, a)$, $\mathbf{w} \in \mathbb{R}^D$ is a weight vector, and $\langle \cdot, \cdot \rangle$ denotes the standard vector dot product.

- **Ranking perceptron** by stochastic subgradient descent:

$$\nabla l_j(\mathbf{w}) = \begin{cases} -\bar{\mathbf{x}}_j & \text{if } \langle \mathbf{w}, \bar{\mathbf{x}}_j \rangle \leq 0, \\ 0 & \text{else.} \end{cases}$$

OL framework: Margin Perceptron

- Hinge loss-type objective

$$l_j(\mathbf{w}) = (1 - \langle \mathbf{w}, \bar{\mathbf{x}}_j \rangle)_+$$

- Stochastic subgradient descent:

$$\nabla l_j(\mathbf{w}) = \begin{cases} -\bar{\mathbf{x}}_j & \text{if } \langle \mathbf{w}, \bar{\mathbf{x}}_j \rangle < 1, \\ 0 & \text{else.} \end{cases}$$

- Margin term controls capacity, but results in more updates.
- Collobert & Bengio (ICML'04) argue that this justifies not using an explicit regularization (as for example in an SGD version of the SVM (Shalev-Shwartz et al. ICML'07)).

OL framework: Margin Perceptron

- Hinge loss-type objective

$$l_j(\mathbf{w}) = (1 - \langle \mathbf{w}, \bar{\mathbf{x}}_j \rangle)_+$$

- Stochastic subgradient descent:

$$\nabla l_j(\mathbf{w}) = \begin{cases} -\bar{\mathbf{x}}_j & \text{if } \langle \mathbf{w}, \bar{\mathbf{x}}_j \rangle < 1, \\ 0 & \text{else.} \end{cases}$$

- Margin term controls capacity, but results in more updates.
- Collobert & Bengio (ICML'04) argue that this justifies not using an explicit regularization (as for example in an SGD version of the SVM (Shalev-Shwartz et al. ICML'07)).

OL framework: Margin Perceptron

- Hinge loss-type objective

$$l_j(\mathbf{w}) = (1 - \langle \mathbf{w}, \bar{\mathbf{x}}_j \rangle)_+$$

- Stochastic subgradient descent:

$$\nabla l_j(\mathbf{w}) = \begin{cases} -\bar{\mathbf{x}}_j & \text{if } \langle \mathbf{w}, \bar{\mathbf{x}}_j \rangle < 1, \\ 0 & \text{else.} \end{cases}$$

- Margin term controls capacity, but results in more updates.
- Collobert & Bengio (ICML'04) argue that this justifies not using an explicit regularization (as for example in an SGD version of the SVM (Shalev-Shwartz et al. ICML'07)).

MTL Framework: ℓ_1/ℓ_2 Block Norm Regularization

- Data points $\{(x_{zn}, y_{zn}), z = 1, \ldots, Z,\ n = 1, \ldots, N_z\}$, sampled from P_z on $X \times Y$ (z = task; n = data point).
- Objective:

$$\min_{\mathbf{W}} \sum_{z,n} l_n(\mathbf{w}_z) + \lambda \|\mathbf{W}\|_{1,2}$$

  - where $\mathbf{W} = (\mathbf{w}_z^d)_{z,d}$ is a Z-by-D matrix $\mathbf{W} = (\mathbf{w}_z^d)_{z,d}$ of D-dimensional row vectors $\mathbf{w}_z$ and Z-dimensional column vectors $\mathbf{w}^d$ of weights associated with feature d across tasks.
- Weighted ℓ_1/ℓ_2 norm:

$$\lambda \|\mathbf{W}\|_{1,2} = \lambda \sum_{d=1}^{D} \|\mathbf{w}^d\|_2$$

  - Each ℓ_2 norm of a weight column $\mathbf{w}^d$ represents the relevance of the corresponding feature across tasks.

MTL Framework: ℓ_1/ℓ_2 Block Norm Regularization

- Data points $\{(x_{zn}, y_{zn}), z = 1, \ldots, Z,\ n = 1, \ldots, N_z\}$, sampled from P_z on $X \times Y$ (z = task; n = data point).
- Objective:

$$\min_{\mathbf{W}} \sum_{z,n} l_n(\mathbf{w}_z) + \lambda \|\mathbf{W}\|_{1,2}$$

  - where $\mathbf{W} = (\mathbf{w}_z^d)_{z,d}$ is a Z-by-D matrix $\mathbf{W} = (\mathbf{w}_z^d)_{z,d}$ of D-dimensional row vectors $\mathbf{w}_z$ and Z-dimensional column vectors $\mathbf{w}^d$ of weights associated with feature d across tasks.
- Weighted ℓ_1/ℓ_2 norm:

$$\lambda \|\mathbf{W}\|_{1,2} = \lambda \sum_{d=1}^{D} \|\mathbf{w}^d\|_2$$

  - Each ℓ_2 norm of a weight column $\mathbf{w}^d$ represents the relevance of the corresponding feature across tasks.

MTL Framework: ℓ_1/ℓ_2 Block Norm Regularization

- Data points $\{(x_{zn}, y_{zn}), z = 1, \ldots, Z,\ n = 1, \ldots, N_z\}$, sampled from P_z on $X \times Y$ (z = task; n = data point).
- Objective:

$$\min_{\mathbf{W}} \sum_{z,n} l_n(\mathbf{w}_z) + \lambda \|\mathbf{W}\|_{1,2}$$

  - where $\mathbf{W} = (\mathbf{w}_z^d)_{z,d}$ is a Z-by-D matrix $\mathbf{W} = (\mathbf{w}_z^d)_{z,d}$ of D-dimensional row vectors $\mathbf{w}_z$ and Z-dimensional column vectors $\mathbf{w}^d$ of weights associated with feature d across tasks.
- Weighted ℓ_1/ℓ_2 norm:

$$\lambda \|\mathbf{W}\|_{1,2} = \lambda \sum_{d=1}^{D} \|\mathbf{w}^d\|_2$$

  - Each ℓ_2 norm of a weight column $\mathbf{w}^d$ represents the relevance of the corresponding feature across tasks.

ℓ_1/ℓ_2 Regularization Explained

	$\mathbf{w}^1$	$\mathbf{w}^2$	$\mathbf{w}^3$	$\mathbf{w}^4$	$\mathbf{w}^5$		$\mathbf{w}^1$	$\mathbf{w}^2$	$\mathbf{w}^3$	$\mathbf{w}^4$	$\mathbf{w}^5$	
$\mathbf{w}_{z_1}$	[6	4	0	0	0]		[6	4	0	0	0]	
$\mathbf{w}_{z_2}$	[0	0	3	0	0]		[3	0	0	0	0]	
$\mathbf{w}_{z_3}$	[0	0	0	2	3]		[2	3	0	0	0]	
column ℓ_2 norm:	6	4	3	2	3		7	5	0	0	0	
ℓ_1 sum:					$\Rightarrow$	18					$\Rightarrow$	12

- ℓ_1 sum of ℓ_2 norms encourages several feature columns $\mathbf{w}^d$ to be **0** and others to have high weights across tasks.
- **Algorithm idea**:
  - Contribution to loss reduction must outweigh regularizer penalty in order to activate feature by non-zero weight.
  - Weight-based feature elimination criterion:

$$\text{If } \|\mathbf{w}^d\|_2 \leq \lambda, \text{ set } \mathbf{W}[z][d] = 0, \forall z.$$

  - Implementation by threshold on K features or by threshold λ.

ℓ_1/ℓ_2 Regularization Explained

$$
\begin{array}{r|ccccc|c|ccccc|c}
 & \mathbf{w}^1 & \mathbf{w}^2 & \mathbf{w}^3 & \mathbf{w}^4 & \mathbf{w}^5 & & \mathbf{w}^1 & \mathbf{w}^2 & \mathbf{w}^3 & \mathbf{w}^4 & \mathbf{w}^5 & \\
\mathbf{w}_{z_1} & [\ 6 & 4 & 0 & 0 & 0\ ] & & [\ 6 & 4 & 0 & 0 & 0\ ] & \\
\mathbf{w}_{z_2} & [\ 0 & 0 & 3 & 0 & 0\ ] & & [\ 3 & 0 & 0 & 0 & 0\ ] & \\
\mathbf{w}_{z_3} & [\ 0 & 0 & 0 & 2 & 3\ ] & & [\ 2 & 3 & 0 & 0 & 0\ ] & \\
\text{column } \ell_2 \text{ norm:} & 6 & 4 & 3 & 2 & 3 & & 7 & 5 & 0 & 0 & 0 & \\
\ell_1 \text{ sum:} & & & & & \Rightarrow & 18 & & & & & \Rightarrow & 12
\end{array}
$$

- ℓ_1 sum of ℓ_2 norms encourages several feature columns $\mathbf{w}^d$ to be **0** and others to have high weights across tasks.
- **Algorithm idea**:
  - Contribution to loss reduction must outweigh regularizer penalty in order to activate feature by non-zero weight.
  - Weight-based feature elimination criterion:

$$\text{If } \|\mathbf{w}^d\|_2 \leq \lambda, \text{ set } \mathbf{W}[z][d] = 0, \forall z.$$

  - Implementation by threshold on K features or by threshold λ.

Multi-Task Learning Algorithm

Algorithm 1 Multi-task Distributed SGD

Get data for Z tasks, each including S sentences;
distribute to machines.
Initialize $\mathbf{v} \leftarrow \mathbf{0}$.
for epochs $t \leftarrow 0 \ldots T-1$: **do**
  for all tasks $z \in \{1 \ldots Z\}$: **parallel do**
    Perform task-specific learning
  end for
  Stack weights $\mathbf{W} \leftarrow [\mathbf{w}_{1,t,S,0}| \ldots |\mathbf{w}_{Z,t,S,0}]^T$
  Perform ℓ_1/ℓ_2 regularization
end for
return v

Implementation as Feature Selection Algorithm

Algorithm 2 Multi-task Distributed SGD

Get data for Z tasks, each including S sentences;
distribute to machines.
Initialize $\mathbf{v} \leftarrow \mathbf{0}$.
for epochs $t \leftarrow 0 \ldots T-1$: **do**
  for all tasks $z \in \{1 \ldots Z\}$: **parallel do**
    $\mathbf{w}_{z,t,0,0} \leftarrow \mathbf{v}$
    for all sentences $i \in \{0 \ldots S-1\}$: **do**
      Decode i^{th} input with $\mathbf{w}_{z,t,i,0}$.
      for all pairs $j \in \{0 \ldots P-1\}$: **do**
        $\mathbf{w}_{z,t,i,j+1} \leftarrow \mathbf{w}_{z,t,i,j} - \eta \nabla l_j(\mathbf{w}_{z,t,i,j})$
      end for
      $\mathbf{w}_{z,t,i+1,0} \leftarrow \mathbf{w}_{z,t,i,P}$
    end for
  end for
  Stack weights $\mathbf{W} \leftarrow [\mathbf{w}_{1,t,S,0}| \ldots |\mathbf{w}_{Z,t,S,0}]^T$
  Select top K feature columns of $\mathbf{W}$ by ℓ_2 norm
  for $k \leftarrow 1 \ldots K$ **do**
    $\mathbf{v}[k] = \frac{1}{Z} \sum_{z=1}^{Z} \mathbf{W}[z][k]$
  end for
end for
return v

Implementation as Adaptive Path-Following Algorithm

Algorithm 3 Path-Following Multi-task Distributed SGD

Get data for Z tasks, each including S sentences; distribute to machines.
Initialize $\mathbf{v} \leftarrow \mathbf{0}$; λ_0, $\lambda_{\min}$, ϵ.
for epochs $t \leftarrow 0 \ldots T-1$: **do**
  for all **tasks** $z \in \{1 \ldots Z\}$: **parallel do**
    Perform task-specific learning
  end for
  Stack weights $\mathbf{W} \leftarrow [\mathbf{w}_{1,t,S,0}| \ldots |\mathbf{w}_{Z,t,S,0}]^T$
  for feature columns $d \in \{1 \ldots D\}$ in $\mathbf{W}$: **do**
    if $\|\mathbf{w}^d\|_2 \leq \lambda_t$ **then**
      $\mathbf{v}[d] = 0$
    else

$$\mathbf{v}[d] = \frac{1}{Z} \sum_{z=1}^{Z} \mathbf{W}[z][d]$$

    end if
  end for
  Set $\lambda_{t+1} = \min\{\lambda_t, \frac{\sum_{z,i,j}(l_{z,i,j}(\mathbf{v}_{t-1}) - l_{z,i,j}(\mathbf{v}_t))}{\epsilon}\}$
  if $\lambda_{t+1} < \lambda_{\min}$ **then**
    break
  end if
end for
return v

SMT using Synchronous Context-Free Grammars

$$
\begin{aligned}
&(1)\ X \rightarrow\ X_1 \text{ hat } X_2 \text{ versprochen; } X_1 \text{ promised } X_2\\
&(2)\ X \rightarrow\ X_1 \text{ hat mir } X_2 \text{ versprochen;}\\
&\qquad\qquad X_1 \text{ promised me } X_2\\
&(3)\ X \rightarrow\ X_1 \text{ versprach } X_2;\ X_1 \text{ promised } X_2
\end{aligned}
$$

- Hierarchical phrase-based translation (Chiang CL'07), formalizes translation rules as productions of synchronous context-free grammar (SCFG).
- Features in discriminative training:
  - **Rule identifiers** for SCFG productions
    Examples: rule (1), (2) and (3)
  - **Rule n-gram** features in source and target
    Examples: "X hat", "hat X", "X versprochen"
  - **Rule shape** features
    Examples: (NT, term$*$, NT, term$*$; NT, term$*$, NT) for (1), (2); (NT, term$*$, NT; NT, term$*$, NT) for rule (3).

SMT using Synchronous Context-Free Grammars

$$
\begin{aligned}
(1)\ X \rightarrow\ & X_1 \text{ hat } X_2 \text{ versprochen; } X_1 \text{ promised } X_2 \\
(2)\ X \rightarrow\ & X_1 \text{ hat mir } X_2 \text{ versprochen;} \\
& \quad X_1 \text{ promised me } X_2 \\
(3)\ X \rightarrow\ & X_1 \text{ versprach } X_2\text{; } X_1 \text{ promised } X_2
\end{aligned}
$$

- Hierarchical phrase-based translation (Chiang CL'07), formalizes translation rules as productions of synchronous context-free grammar (SCFG).
- Features in discriminative training:
  - **Rule identifiers** for SCFG productions
    Examples: rule (1), (2) and (3)
  - **Rule n-gram** features in source and target
    Examples: "X hat", "hat X", "X versprochen"
  - **Rule shape** features
    Examples: (NT, term$*$, NT, term$*$; NT, term$*$, NT) for (1), (2); (NT, term$*$, NT; NT, term$*$, NT) for rule (3).

Experiment I: Random Sharding on Large Parallel Data

- **Idea:** Take advantage of inherent efficiency (and effectiveness) of multi-task learning.
  - Define **tasks as random shards** of data,
  - either by **sharding once** or by **re-sharding** after each epoch.
- Advantage:
  - Hadoop/MapReduce framework offers parallelization by data sharding.
  - Feature selection by ℓ_1/ℓ_2 block norm regularization on shards iteratively cuts feature space to feasible size.
- See Simianer, Riezler, Dyer ACL'12.

Experiment I: Random Sharding on Large Parallel Data

- **Idea:** Take advantage of inherent efficiency (and effectiveness) of multi-task learning.
  - Define **tasks as random shards** of data,
  - either by **sharding once** or by **re-sharding** after each epoch.
- Advantage:
  - Hadoop/MapReduce framework offers parallelization by data sharding.
  - Feature selection by ℓ_1/ℓ_2 block norm regularization on shards iteratively cuts feature space to feasible size.
- See Simianer, Riezler, Dyer ACL'12.

Experiment I: Random Sharding on Large Parallel Data

- **Idea:** Take advantage of inherent efficiency (and effectiveness) of multi-task learning.
  - Define **tasks as random shards** of data,
  - either by **sharding once** or by **re-sharding** after each epoch.
- Advantage:
  - Hadoop/MapReduce framework offers parallelization by data sharding.
  - Feature selection by ℓ_1/ℓ_2 block norm regularization on shards iteratively cuts feature space to feasible size.
- See Simianer, Riezler, Dyer ACL'12.

Data

News Commentary(*nc*)

	train-*nc*	**lm-train-*nc***	**dev-*nc***	**devtest-*nc***	**test-*nc***
Sentences	132,753	180,657	1057	1064	2007
Tokens *de*	3,530,907	–	27,782	28,415	53,989
Tokens *en*	3,293,363	4,394,428	26,098	26,219	50,443
Rule Count	14,350,552 (1G)	–	2,322,912	2,320,264	3,274,771

Europarl(*ep*)

	train-*ep*	**lm-train-*ep***	**dev-*ep***	**devtest-*ep***	**test-*ep***
Sentences	1,655,238	2,015,440	2000	2000	2000
Tokens *de*	45,293,925	–	57,723	56,783	59,297
Tokens *en*	45,374,649	54,728,786	58,825	58,100	60,240
Rule Count	203,552,525 (31.5G)	–	17,738,763	17,682,176	18,273,078

News Crawl(*crawl*)

		dev-*crawl*	**test-*crawl10***	**test-*crawl11***
Sentences		2051	2489	3003
Tokens *de*		49,848	64,301	76,193
Tokens *en*		49,767	61,925	74,753
Rule Count		9,404,339	11,307,304	12,561,636

SMT Setup

- cdec (Dyer et al. ACL'10) framework for decoding and induction of SCFGs.
- SCFG per-sentence grammars are stored on disk instead of in memory (Lopez EMNLP'07), extracted by leave-one-out (Zollmann and Sima'an JACL'05) for training-set tuning.
- Scale:
    - Data are split into shards holding about 1,000 sentences, corresponding to dev set size.
    - On Hadoop/MapReduce cluster for 300 parallel jobs this required 2,290 shards for *ep* data set.
    - 5M active features without feature selection on *ep* data set.

SMT Setup

- `cdec` (Dyer et al. ACL'10) framework for decoding and induction of SCFGs.
- SCFG per-sentence grammars are stored on disk instead of in memory (Lopez EMNLP'07), extracted by leave-one-out (Zollmann and Sima'an JACL'05) for training-set tuning.
- Scale:
  - Data are split into shards holding about 1,000 sentences, corresponding to dev set size.
  - On Hadoop/MapReduce cluster for 300 parallel jobs this required 2,290 shards for *ep* data set.
  - 5M active features without feature selection on *ep* data set.

SMT Setup

- `cdec` (Dyer et al. ACL'10) framework for decoding and induction of SCFGs.
- SCFG per-sentence grammars are stored on disk instead of in memory (Lopez EMNLP'07), extracted by leave-one-out (Zollmann and Sima'an JACL'05) for training-set tuning.
- Scale:
    - Data are split into shards holding about 1,000 sentences, corresponding to dev set size.
    - On Hadoop/MapReduce cluster for 300 parallel jobs this required 2,290 shards for *ep* data set.
    - 5M active features without feature selection on *ep* data set.

Results on News Commentary (*nc*) data

Algorithm	Tuning set	Features	#Features	test-*nc*
Single-task SGD	dev-*nc*	default	12	28.0
	dev-*nc*	+id,ng,shape	180k	28.15
Multi-task SGD	train-*nc*	+id,ng,shape	100k	**28.81**

- Scaling from 12 to 180K features on dev set does not help.
- Scaling to full feature- and training-set does help (+0.8 BLEU).
- Statistical significance assessed by Approximate Randomization (Noreen'89).

Results on Europarl (*ep*) and News Crawl (*crawl*) data

Algorithm	Tuning set	Features	#Features	test-*ep*
Single-task SGD	dev-*ep*	default	12	26.42
	dev-*ep*	+id,ng,shape	300k	**28.37**
Multi-task SGD	train-*ep*	+id,ng,shape	100k	**28.62**

Alg.	Tuning set	Features	#Feats	test-*crawl*10	test-*crawl*11
ST	dev-*crawl*	default	12	15.39	14.43
	dev-*crawl*	+id,ng,shape	300k	**17.8**	**16.83**
MT	train-*ep*	+id,ng,shape	100k	**19.12**	**17.33**

- Scaling up feature sets helps even for dev-set tuning.
- On large scale tuning set only Multi-task SGD is feasible.
- Additional gains of 0.5 to 1.3 BLEU by scaling to large tuning set on out-of-domain news crawl test data.

Experiments II: Random vs. Natural Tasks

- **Research Question**:
  - As shown, multi-task learning can be used as general regularization technique on random shards.
  - Can multi-task learning benefit from **natural task structure** in the data, where shared and individual knowledge is properly balanced?
- See Simianer & Riezler WMT'13.

Experiments II: Random vs. Natural Tasks

- **Research Question**:
  - As shown, multi-task learning can be used as general regularization technique on random shards.
  - Can multi-task learning benefit from **natural task structure** in the data, where shared and individual knowledge is properly balanced?
- See Simianer & Riezler WMT'13.

Experiments II: Random vs. Natural Tasks

- **Research Question**:
  - As shown, multi-task learning can be used as general regularization technique on random shards.
  - Can multi-task learning benefit from **natural task structure** in the data, where shared and individual knowledge is properly balanced?
- See Simianer & Riezler WMT'13.

Data

A	Human Necessities
B	Performing Operations, Transporting
C	Chemistry, Metallurgy
D	Textiles, Paper
E	Fixed Constructions
F	Mechanical Engineering, Lighting, Heating, Weapons
G	Physics
H	Electricity

- International Patent Classification (IPC) categorizes patents hierarchically into eight sections, 120 classes, 600 subclasses, down to 70,000 subgroups at the leaf level.
- Typically, a patent belongs to more than one section, with one section chosen as main classification.
- Eight top classes/sections used to define **natural tasks**.

SMT and Learning Setup

- SCFG framework using sparse local features (as above).
- Learning algorithms:
  - Baselines:
    - MERT (Kumar et al. ACL'09)
    - Single-task perceptron w/ and w/o ℓ_1 regularization with clipping (Carpenter 2008)
    - Single-task margin perceptron (Collobert & Bengio ICML'04).
  - Multi-task tuning using standard and margin perceptron.
  - Tuning methods with random components (MERT, random (re)sharding) were repeated 3 times and BLEU scores averaged.

SMT and Learning Setup

- SCFG framework using sparse local features (as above).
- Learning algorithms:
  - Baselines:
    - MERT (Kumar et al. ACL'09)
    - Single-task perceptron w/ and w/o ℓ_1 regularization with clipping (Carpenter 2008)
    - Single-task margin perceptron (Collobert & Bengio ICML'04).
  - Multi-task tuning using standard and margin perceptron.
  - Tuning methods with random components (MERT, random (re)sharding) were repeated 3 times and BLEU scores averaged.

Train/dev/test splits

- 1.2M parallel sentences from patent domain for training[1].
- Development and test sets of 2,000 sentences from each of sections A to H for **independent** tuning and testing.
- **Pooled** development and test sets containing 2,000 sentences with all sections evenly represented.
- **Pooled-cat** development set for tuning on concatenation of data from all sections.

[1] `http://www.cl.uni-heidelberg.de/statnlpgroup/pattr`

Train/dev/test splits

- 1.2M parallel sentences from patent domain for training[1].
- Development and test sets of 2,000 sentences from each of sections A to H for **independent** tuning and testing.
- **Pooled** development and test sets containing 2,000 sentences with all sections evenly represented.
- **Pooled-cat** development set for tuning on concatenation of data from all sections.

[1] `http://www.cl.uni-heidelberg.de/statnlpgroup/pattr`

MERT Baseline w/ 12 Dense Features

	single-task tuning		
	indep. ^{0}	pooled ^{1}	pooled-cat ^{2}
pooled test	–	51.18	51.22
A	54.92	^{02}55.27	^{0}55.17
B	51.53	51.48	^{01}51.69
C	^{12}56.31	^{2}55.90	55.74
D	49.94	^{0}50.33	^{0}50.26
E	^{1}49.19	48.97	^{1}49.13
F	^{12}51.26	51.02	51.12
G	^{1}49.61	49.44	49.55
H	49.38	49.50	^{01}49.67
average test	**51.52**	**51.49**	**51.54**

- Neither tuning on *pooled* or *pooled-cat* improves over *indep.*.
- $^{x \subset \{0,1,2\}}$BLEU denotes statistical significance of pairwise test.

Single-Task Perceptron w/ ℓ_1 Regularization

	single-task tuning		
	indep. [0]	pooled [1]	pooled-cat [2]
pooled test	–	50.75	[1] 52.08
A	[1] 55.11	54.32	[01] 55.94
B	[1] 52.61	50.84	[1] 52.57
C	56.18	56.11	[01] 56.75
D	[1] 50.68	49.48	[01] 51.22
E	[1] 50.27	48.69	[1] 50.01
F	[1] 51.68	50.71	[1] 51.95
G	[1] 49.90	49.06	[01] 50.51
H	[1] 50.48	49.16	[1] 50.53
average test	**52.11**	**51.05**	**52.44**
model size	430,092.5	457,428	1,574,259

- Improvements over MERT, mostly on *pooled-cat* tuning set.
- 1.5M features make serial tuning on *pooled-cat* infeasible.
- Overfitting effect on small *pooled* data.

Single- and Multi-Task Perceptron

	single-task tuning			multi-task tuning		
	indep. ^{0}	pooled ^{1}	pooled-cat ^{2}	IPC ^{3}	sharding ^{4}	resharding ^{5}
pooled test	–	51.33	1 51.77	^{12} 52.56	^{12} 52.54	^{12} 52.60
A	54.79	54.76	^{01} 55.31	^{012} 56.35	^{012} 56.22	^{012} 56.21
B	^{12} 52.45	51.30	^{1} 52.19	^{012} 52.78	^{0123} 52.98	^{012} 52.96
C	^{2} 56.62	56.65	^{1} 56.12	^{01245} 57.76	^{012} 57.30	^{012} 57.44
D	^{1} 50.75	49.88	^{1} 50.63	^{01245} 51.54	^{012} 51.33	^{012} 51.20
E	^{1} 49.70	49.23	^{01} 49.92	^{012} 50.51	^{012} 50.52	^{012} 50.38
F	^{1} 51.60	51.09	^{1} 51.71	^{012} 52.28	^{012} 52.43	^{012} 52.32
G	^{1} 49.50	49.06	^{01} 49.97	^{012} 50.84	^{012} 50.88	^{012} 50.74
H	^{1} 49.77	49.50	^{01} 50.64	^{012} 51.16	^{012} 51.07	^{012} 51.10
average test	**51.90**	**51.42**	**52.06**	**52.90**	**52.84**	**52.79**
model size	366,869.4	448,359	1,478,049	100,000	100,000	100,000

- Multi-task tuning improves BLEU over all single-task runs.
- Also more efficient due to iterative feature selection.
- Difference between natural and random tasks inconclusive.

Single- and Multi-Task Margin Perceptron

	single-task tuning			multi-task tuning		
	indep. [0]	pooled [1]	pooled-cat [2]	IPC [3]	sharding [4]	resharding [5]
pooled test	–	51.33	[1] 52.58	[12] 52.98	[12] 52.95	[12] 52.99
A	[1] 56.09	55.33	[1] 55.92	[01245] 56.78	[012] 56.62	[012] 56.53
B	[1] 52.45	51.59	[1] 52.44	[012] 53.31	[012] 53.35	[012] 53.21
C	[1] 57.20	56.85	[01] 57.54	[01] 57.46	[1] 57.42	[1] 57.43
D	[1] 50.51	50.18	[01] 51.38	[01245] 52.14	[0125] 51.82	[012] 51.66
E	[1] 50.27	49.36	[01] 50.72	[0124] 51.13	[012] 50.89	[012] 51.02
F	[1] 52.06	51.20	[01] 52.61	[01245] 53.07	[012] 52.80	[012] 52.87
G	[1] 50.00	49.58	[01] 50.90	[01245] 51.36	[012] 51.19	[012] 51.11
H	[1] 50.57	49.80	[01] 51.32	[012] 51.57	[012] 51.62	[01] 51.47
average test	**52.39**	**51.74**	**52.85**	**53.35**	**53.21**	**53.16**
model size	423,731.5	484,483	1,697,398	100,000	100,000	100,000

- Single-task runs beat standard perceptron w/ and w/o ℓ_1.
- Regularization by margin and multi-task learning adds up.
- Best result is nearly 2 BLEU points better than MERT.

Conclusion

- Multi-task learning for SMT is **efficient** due to online learning, parallelization and feature selection,
- but also **effective** in terms of BLEU improvements over single-task learning.
- Multi-task learning is **adaptive** due to path-following in regularization.
- Question: Can **task definition be adapted to problem** as well?
  - *Natural* task definition show nominal (not statistically significant) advantage.
  - Future work: Optimize clustering of IPC subclasses for multi-task learning in SMT.

Conclusion

- Multi-task learning for SMT is **efficient** due to online learning, parallelization and feature selection,
- but also **effective** in terms of BLEU improvements over single-task learning.
- Multi-task learning is **adaptive** due to path-following in regularization.
- Question: Can **task definition be adapted to problem** as well?
  - *Natural* task definition show nominal (not statistically significant) advantage.
  - Future work: Optimize clustering of IPC subclasses for multi-task learning in SMT.

Conclusion

- Multi-task learning for SMT is **efficient** due to online learning, parallelization and feature selection,
- but also **effective** in terms of BLEU improvements over single-task learning.
- Multi-task learning is **adaptive** due to path-following in regularization.
- Question: Can **task definition be adapted to problem** as well?
  - *Natural* task definition show nominal (not statistically significant) advantage.
  - Future work: Optimize clustering of IPC subclasses for multi-task learning in SMT.

Conclusion

- Multi-task learning for SMT is **efficient** due to online learning, parallelization and feature selection,
- but also **effective** in terms of BLEU improvements over single-task learning.
- Multi-task learning is **adaptive** due to path-following in regularization.
- Question: Can **task definition be adapted to problem** as well?
  - *Natural* task definition show nominal (not statistically significant) advantage.
  - Future work: Optimize clustering of IPC subclasses for multi-task learning in SMT.

Conclusion

- Multi-task learning for SMT is **efficient** due to online learning, parallelization and feature selection,
- but also **effective** in terms of BLEU improvements over single-task learning.
- Multi-task learning is **adaptive** due to path-following in regularization.
- Question: Can **task definition be adapted to problem** as well?
  - *Natural* task definition show nominal (not statistically significant) advantage.
  - Future work: Optimize clustering of IPC subclasses for multi-task learning in SMT.

IPC

IPC: 8 sections, 120 classes, 600 subclasses, 70,000 subgroups:
Is there a *natural* or *useful* task definition for multi-task SMT?

Code

- `dtrain` code is part of `cdec`:
  `https://github.com/redpony/cdec`.

Thanks for your attention!

Phil Blunsom, Trevor Cohn, and Miles Osborne.
A discriminative latent variable model for statistical machine translation.
In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT'08)*, Columbus, OH, 2008.

Leon Bottou and Olivier Bousquet.
Large scale online learning.
In *Proceedings of the 18th Annual Conference on Neural Information Processing System (NIPS'04)*, Vancouver, Canada, 2004.

Leon Bottou and Olivier Bousquet.
The tradeoffs of large scale learning.
In *Proceedings of the 21st Annual Conference on Neural Information Processing System (NIPS'07)*, Vancouver, Canada, 2007.

Leo Breiman.
Bagging predictors.
Machine Learning, 24:123–140, 1996.

Bob Carpenter.
Lazy sparse stochastic gradient descent for regularized multinomial logistic regression.
Technical report, Alias-i, 2008.

David Chiang.
Hierarchical phrase-based translation.
Computational Linguistics, 33(2), 2007.

Ronan Collobert and Samy Bengio.
Links between perceptrons, MLPs, and SVMs.
In *Proceedings of the 21st International Conference on Machine Learning (ICML'04)*, Banff, Canada, 2004.

Chris Dyer, Adam Lopez, Juri Ganitkevitch, Jonathan Weese, Ferhan Ture, Phil Blunsom, Hendra Setiawan, Vladimir Eidelman, and Philip Resnik.
cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models.
In *Proceedings of the ACL 2010 System Demonstrations*, Uppsala, Sweden, 2010.

Jeffrey Flanigan, Chris Dyer, and Jaime Carbonell.
Large-scale discriminative training for statistical machine translation using held-out line search.
In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT'13)*, Atlanta, GA, 2013.

Shankar Kumar, Wolfgang Macherey, Chris Dyer, and Franz Och.

Efficient minimum error rate training and minimum Bayes-risk decoding for translation hypergraphs and lattices.
In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th IJCNLP of the AFNLP (ACL-IJCNLP'09*, Suntec, Singapore, 2009.

John Langford, Lihong Li, and Tong Zhang.
Sparse online learning via truncated gradient.
Journal of Machine Learning Research, 10:777–801, 2009.

Percy Liang, Alexandre Bouchard-Côté, Dan Klein, and Ben Taskar.
An end-to-end discriminative approach to machine translation.
In *Proceedings of the joint conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics (COLING-ACL'06)*, Sydney, Australia, 2006.

Adam Lopez.

Hierarchical phrase-based translation with suffix arrays.
In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007)*, Prague, Czech Republic, 2007.

André F.T. Martins, Noah A. Smith, Pedro M.Q.Aguiar, and Mário A.T. Figueiredo.
Structured sparsity in structured prediction.
In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, Edinburgh, Scotland, 2011.

Ryan McDonald, Keith Hall, and Gideon Mann.
Distributed training strategies for the structured perceptron.
In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT'10)*, Los Angeles, CA, 2010.

Preslav Nakov, Francisco Guzmán, and Stephan Vogel.
Optimizing for sentence-level bleu+1 yields short translations.
In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)*, Bombay, India, 2012.

Eric W. Noreen.
Computer Intensive Methods for Testing Hypotheses. An Introduction.
Wiley, New York, 1989.

Guillaume Obozinski, Ben Taskar, and Michael I. Jordan.
Joint covariate selection and joint subspace selection for multiple classification problems.
Statistics and Computing, 20:231–252, 2010.

Franz Josef Och.
Minimum error rate training in statistical machine translation.

In *Proceedings of the Human Language Technology Conference and the 3rd Meeting of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL'03)*, Edmonton, Cananda, 2003.

Shai Shalev-Shwartz, Yoram Singer, and Nathan Srebro.
Pegasos: Primal Estimated sub-GrAdient SOlver for SVM.
In *Proceedings of the 24th International Conference on Machine Learning (ICML'07)*, Corvallis, OR, 2007.

Libin Shen and Aravind K. Joshi.
Ranking and reranking with perceptron.
Journal of Machine Learning Research, 60(1-3):73–96, 2005.

Patrick Simianer and Stefan Riezler.
Multi-task learning for improved discriminative training in SMT.
In *Proceedings of the ACL 2013 Eighth Workshop on Statistical Machine Translation (WMT'13)*, Sofia, Bulgaria, 2013.

Patrick Simianer, Stefan Riezler, and Chris Dyer.
Joint feature selection in distributed stochastic learning for large-scale discriminative training in SMT.
In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012)*, Jeju, Korea, 2012.

Christoph Tillmann and Tong Zhang.
A discriminative global training algorithm for statistical MT.
In *Proceedings of the joint conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics (COLING-ACL'06)*, Sydney, Australia, 2006.

Ming Yuan and Yi Lin.
Model selection and estimation in regression with grouped variables.
J.R.Statist.Soc.B, 68(1):49–67, 2006.

Peng Zhao, Guilherme Rocha, and Bin Yu.

The composite absolute penalties family for grouped and hierarchical variable selection.
The Annals of Statistics, 37(6A):3468–3497, 2009.

Martin A. Zinkevich, Markus Weimer, Alex Smola, and Lihong Li.
Parallelized stochastic gradient descent.
In *Proceedings of the 24th Annual Conference on Neural Information Processing Sytems (NIPS'10)*, Vancouver, Canada, 2010.

Andreas Zollmann and Khalil Sima'an.
A consistent and efficient estimator for data-oriented parsing.
Journal of Automata, Languages and Combinatorics, 10(2/3):367–388, 2005.