Robust Estimation of Feature Weights in Statistical Machine Translation

Cristina España-Bonet and Lluís Màrquez

Departament de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya

OpenMT-2 Meeting

Donostia, 25th January 2010

Overview

1. Motivation
2. Perceptron-based training
3. Experiments
4. Conclusions

Motivation

SMT, the log-linear model

$$T(f) = \hat{e} = \mathrm{argmax_e} \; \log P(e|f) = \mathrm{argmax_e} \; \sum_m \lambda_m h_m(f|e)$$

Motivation

SMT, the log-linear model

$$T(f) = \hat{e} = \mathrm{argmax_e}\ \log P(e|f) = \mathrm{argmax_e}\ \sum_m \lambda_m h_m(f|e)$$

- $f \longrightarrow$ source, $e \longrightarrow$ target
- $h_m \longrightarrow$ features (log-probabilities)
  - Language and translation models,
  - and distortion, word penalty, phrase penalty...
- $\lambda_m \longrightarrow$ weight of every feature

Motivation

Minimum Error Rate Training, MERT

Weight optimisation:

MERT, an optimisation of the translation performance (usually with BLEU as the reference score)

Motivation

Minimum Error Rate Training, MERT

Weight optimisation:

MERT, an optimisation of the translation performance (usually with BLEU as the reference score)

Common criticisms:

- Limits the number of weights to tens at most
- Possibility of being stuck in a local minimum
- Overfitting

Motivation

Minimum Error Rate Training, MERT

Weight optimisation:

MERT, an optimisation of the translation performance (usually with BLEU as the reference score)

Common criticisms:

- Limits the number of weights to tens at most
- Possibility of being stuck in a local minimum
- Overfitting

Perceptron-based training

The Perceptron

Proposal:

Weight estimation via a perceptron training

Perceptron score function

$$\text{score} = \sum_m \lambda_m h_m(\mathit{input}, \mathit{output})$$

Perceptron-based training

The Perceptron

Proposal:

Weight estimation via a perceptron training

SMT score function

$$\mathrm{score} = \sum_m \lambda_m h_m(\mathit{source}, \mathit{target})$$

Perceptron-based training

The Perceptron

Proposal:

Weight estimation via a perceptron training

Perceptron score function

$$\text{score} = \sum_m \lambda_m h_m(\mathit{source}, \mathit{target})$$

$$\lambda_m \leftarrow \lambda_m \; + \; h_m(\mathit{source}, \mathit{target}) - h_m(\mathit{source}, \mathit{guess})$$

For each training example and N epochs

algorithm

Perceptron-based training
Choices

First main choices:

- Features h_m
  - Probabilities SMT: 8 reals

Perceptron-based training

Choices

First main choices:

- Features h_m
  - Probabilities SMT: 8 reals
- Gold standard
  - Optimise towards the translation with the highest BLEU in an n-best list

Arabic-to-English translation

- Training corpus: 125 ksegments (newswire domain)

Experiments
Translation task, data sets

Arabic-to-English translation

- Training corpus: 125 ksegments (newswire domain)
- Development/Test corpora:

Test set	Segments	Genre	OOVs (%)	Perplexity Arabic	Perplexity English
Trdev	500	newswire	1.25	272	129
Trtest	500	newswire	1.18	270	133
N05	1056	newswire	2.02	320	145
N06	1797	newswire & web	5.16	598	205
N08	1357	newswire & web	3.82	568	227

Experiments

Translation task, data sets

Arabic-to-English translation

- Training corpus: 125 ksegments (newswire domain)
- Development/Test corpora:

Test set	Segments	Genre	OOVs (%)	Perplexity Arabic	Perplexity English
Trdev	500	newswire	1.25	272	129
Trtest	500	newswire	1.18	270	133
N05	1056	newswire	2.02	320	145
N06	1797	newswire & web	5.16	598	205
N08	1357	newswire & web	3.82	568	227

Experiments

Translation task, data sets

Arabic-to-English translation

- Training corpus: 125 ksegments (newswire domain)
- Development/Test corpora:

Test set	Segments	Genre	OOVs (%)	Perplexity Arabic	Perplexity English
Trdev	500	newswire	1.25	272	129
Trtest	500	newswire	1.18	270	133
N05	1056	newswire	2.02	320	145
N06	1797	newswire & web	5.16	598	205
N08	1357	newswire & web	3.82	568	227

Two distinct experiments according to the available data:

1. Domain Adaptation
   training $\sim$ development $\neq$ test

Two distinct experiments according to the available data:

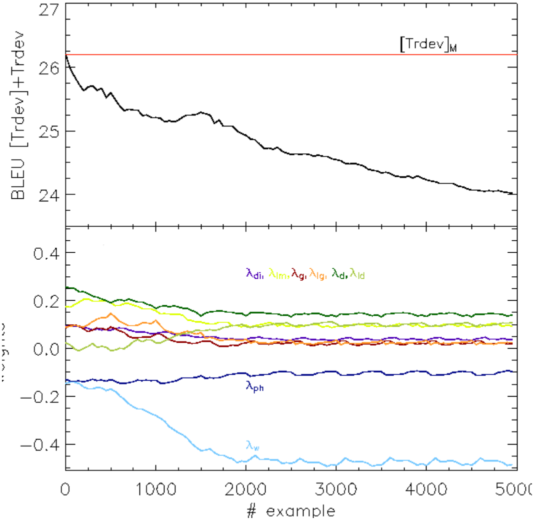
1. Domain Adaptation
   training $\sim$ development $\neq$ test
2. Domain Tuning
   training $\neq$ development $\sim$ test

Two distinct experiments according to the available data:

1. Domain Adaptation
   training $\sim$ development $\neq$ test
2. Domain Tuning
   training $\neq$ development $\sim$ test

Perceptron training

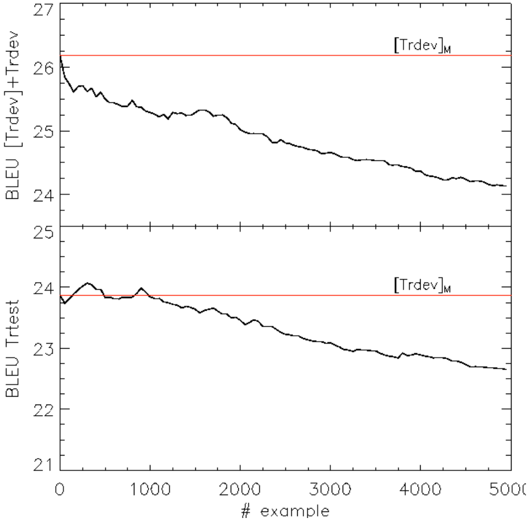
- set *Trdev*
- 500 examples
- MERT λ_0
- N=10 iterations



Experiments
Domain Adaptation, Test

Results on *Trtest* throughout the training

- In-domain test set

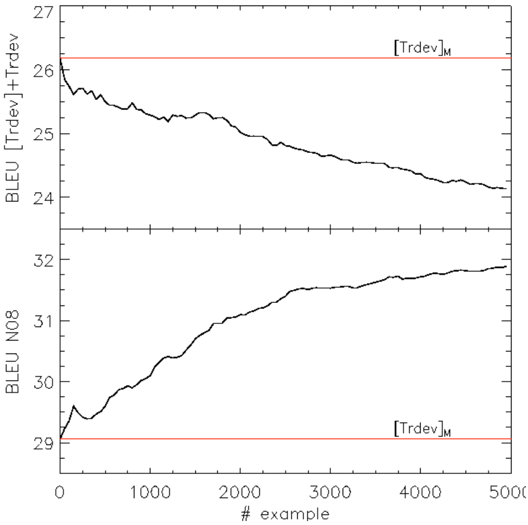


Experiments

Domain Adaptation, Test

Results on *N*08 throughout the training

- Out-of-domain test set



Experiments
Domain Adaptation, Test

Numerical results at the stopping point:

	BLEU			
	Trtest	N05	N06	N08
$[\text{Trdev}]_{\mathrm{M}}$	**23.87**	43.76	30.24	29.06
$[\text{Trdev}]_{\mathrm{M}} + \text{Trdev}$	23.10	**43.90**	**32.08**	**31.48**
MERT on test	24.27	45.46	32.96	32.77

Conclusions

Summary

1. We apply 2 development stages: MERT+Percpt.
2. The first stage, MERT, locates a good point in the weights space for the development set.
3. The second stage, Percpt, generalises the obtained values to be used in test.
4. The combination of both improves $\sim$ 2 BLEU points on out-of-domain tests but worsens on in-domain sets.

Robust Estimation of Feature Weights in Statistical Machine Translation

Cristina España-Bonet and Lluís Màrquez

Departament de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya

OpenMT-2 Meeting

Donostia, 25th January 2010

Averaged perceptron

The algorithm

Input:

Training data, $\{f^i, e^i\}_{i=1}^T$
Initial weights, $\overrightarrow{\lambda}_0$
N epochs, learning rate ϵ

for each example f_i $i = 1, ..., T$
$\hat{\mathbf{e}} = \text{decode}(f_i, \lambda_i)$
guess: $\hat{e}[1]$
target: $\text{argmax}_{\hat{\mathbf{e}}}(\text{BLEU}(\hat{\mathbf{e}}))$
if $\overrightarrow{h}(\text{guess}) \neq \overrightarrow{h}(\text{tgt})$ **then**
$\overrightarrow{\lambda}_i := \overrightarrow{\lambda}_i + \epsilon \cdot \Delta \overrightarrow{h}_i(f_i, \text{tgt}, \text{guess})$
end if
$\overrightarrow{\Lambda} := \overrightarrow{\Lambda} + \overrightarrow{\lambda}_i$
end for

Averaged perceptron
The algorithm

Input:
Training data, $\{f^i, e^i\}_{i=1}^{T}$
Initial weights, $\overrightarrow{\lambda}_0$
N epochs, learning rate ϵ
for each epoch $n = 1, ..., N$
for each example f_i $i = 1, ..., T$
$\hat{\mathbf{e}}$ = decode(f_i,λ_i)
guess: $\hat{e}[1]$
target: $\mathrm{argmax}_{\hat{e}}\,(\mathrm{BLEU}(\hat{\mathbf{e}}))$
if $\overrightarrow{h}(\mathrm{guess}) \neq \overrightarrow{h}(\mathrm{tgt})$ **then**
$\overrightarrow{\lambda}_i := \overrightarrow{\lambda}_i \; + \; \epsilon \cdot \Delta \overrightarrow{h}_i(f_i, \mathrm{tgt}, \mathrm{guess})$
end if
$\overrightarrow{\Lambda} := \overrightarrow{\Lambda} \; + \; \overrightarrow{\lambda}_i$
end for
end for
return $(\overrightarrow{\Lambda}/NT)$

back

Perceptron-based training

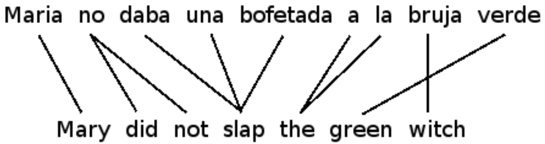
Why not the reference translation being the gold standard?

Maria no daba una bofetada a la bruja verde

Mary did not slap the green witch

Perceptron-based training

Why not the reference translation being the gold standard?

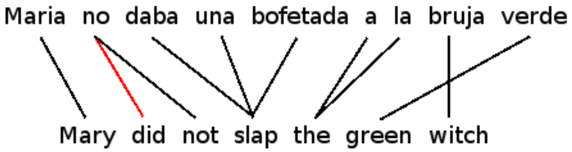



```
(Maria, Mary)
(no, did not)
(slap, daba una bofetada)
(a la, the)
(bruja, witch)
(verde, green)
(Maria no, Mary did not)
(no daba una bofetada, did not slap)
(daba una bofetada a la, slap the)
(bruja verde, green witch)
(Maria no daba una bofetada, Mary did not slap)
(no daba una bofetada a la, did not slap the)
(a la bruja verde, the green witch)
(Maria no daba una bofetada a la, Mary did not slap the)
(daba una bofetada a la bruja verde, slap the green witch)
(no daba una bofetada a la bruja verde, did not slap the green witch)
(Maria no daba una bofetada a la bruja verde, Mary did not slap the green witch)
```

☞ A same translation is reachable through multiple phrase combinations

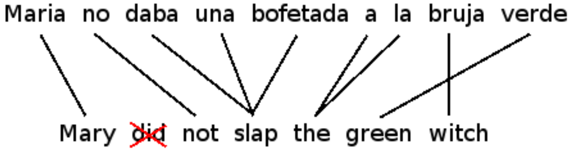
Perceptron-based training

Why not the reference translation being the gold standard?



Perceptron-based training

Why not the reference translation being the gold standard?

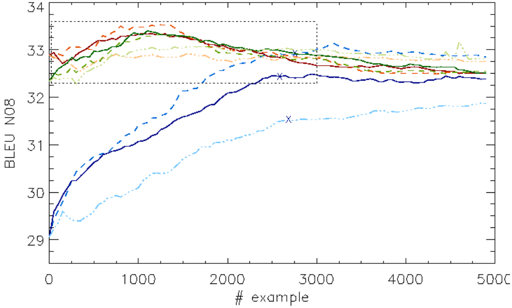


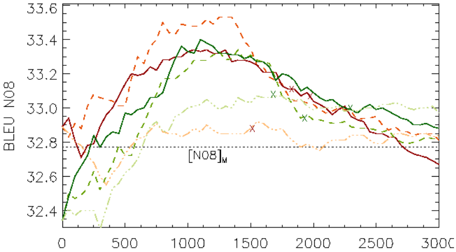
☞ A translation can be NOT reachable through the extracted phrases

Experiments

Domain Tuning, Test

Results on N08 test set throughout the training





Numerical results at the stopping point:

	MERT set		
Perceptron set	$[\text{Trdev}]_{\mathrm{M}}$	$[\text{N06}]_{\mathrm{M}}$	$[\text{Trdev.N06}]_{\mathrm{M}}$
–	29.06	32.89	32.34
Trdev	31.48	32.98	**33.11**
N06	**32.83**	**33.01**	33.05
Trdev.N06	32.46	32.98	33.05