Discriminative Training for Phrase-Based Machine Translation

Abhishek Arun

19 April 2007

Overview

- Evolution from generative to discriminative models
- Discriminative training
- Model
- Learning schemes
- Featured representation
- The reference dilemna
- Experiments
- Future work
- Conclusion

The birth of SMT: generative models

- The definition of translation probability follows a **mathematical derivation**

$$\text{argmax}_{\mathbf{e}} p(\mathbf{e}|\mathbf{f}) = \text{argmax}_{\mathbf{e}} p(\mathbf{f}|\mathbf{e})\ p(\mathbf{e}) \tag{1}$$

- Occasionally, some **independence assumptions** are thrown in
  for instance IBM Model 1: word translations are independent of each other

$$p(\mathbf{e}|\mathbf{f}, a) = \frac{1}{Z} \prod_i p(e_i|f_{a(i)})$$

- Generative model leads to **straight-forward estimation**
  - maximum likelihood estimation of component probability distribution
  - **EM algorithm** for discovering hidden variables (alignment)

Log-linear models

- Alternative to Equation 1 : Model **posterior probability directly** :

$$\mathsf{p}(\mathbf{e}|\mathbf{f}) = \frac{exp[\sum_{m=1}^{M} \lambda_m h_m(\mathbf{e},\mathbf{f})]}{\sum_{e'} exp[\sum_{m=1}^{M} \lambda_m h_m(\mathbf{e'},\mathbf{f})]} \tag{2}$$

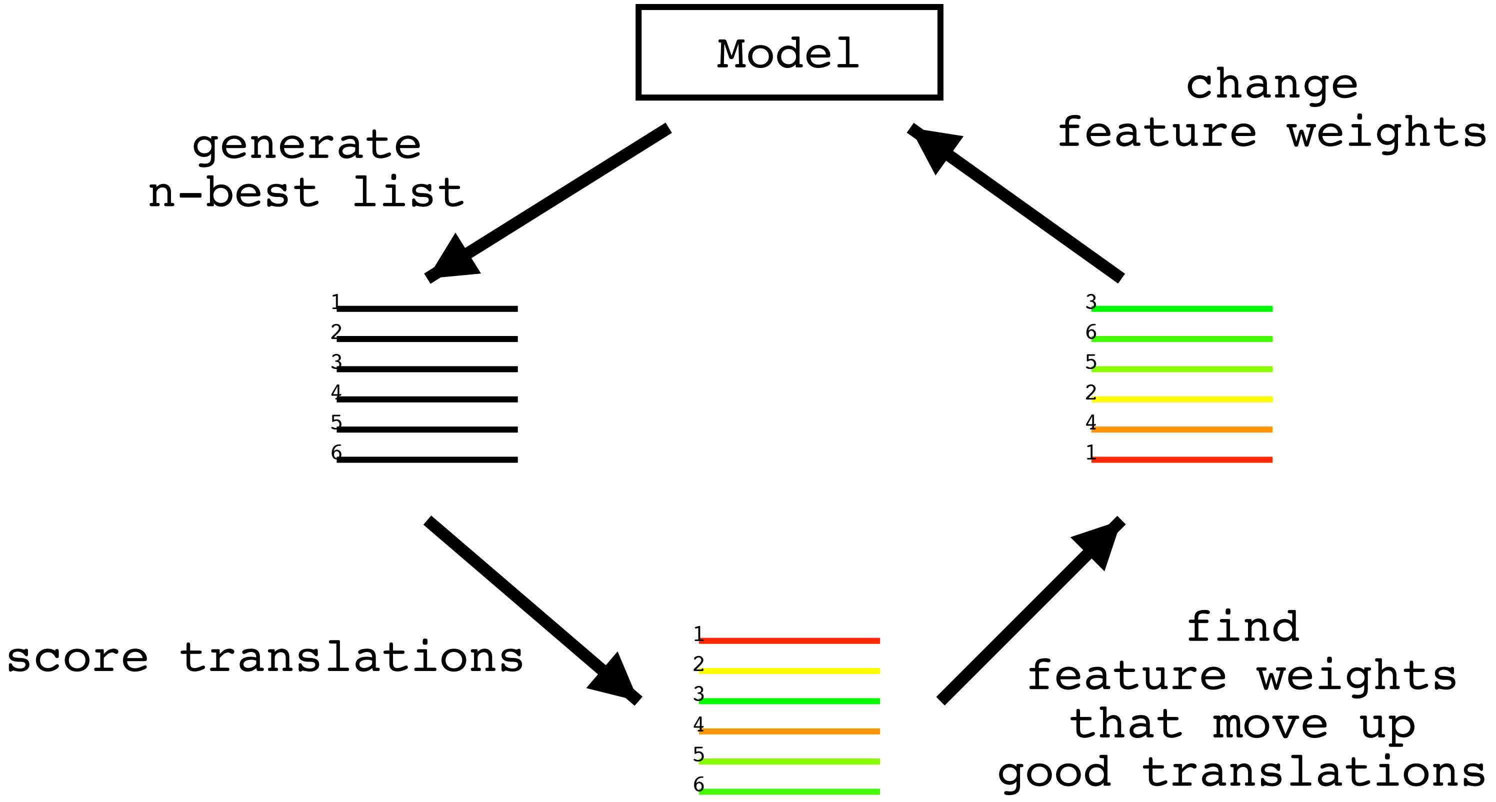
- Decision rule is now :

$$\begin{aligned} \hat{e} &= \mathsf{argmax}_{\mathbf{e}} p(\mathbf{e}|\mathbf{f}) \\ &= \mathsf{argmax}_{\mathbf{e}}[\sum_{m=1}^{M} \lambda_m h_m(\mathbf{e},\mathbf{f})] \end{aligned}$$

Discriminative training

- **Modeling problem**:
  - Come up with sensible features.
- **Training problem**:
  - Come up with suitable lambdas.
- Most estimation procedures in NLP maximize likelihood of training data.
- However at test time model is evaluated wrt to some **loss function**
- Idea:
  - Minimize loss on training data

Discriminative training



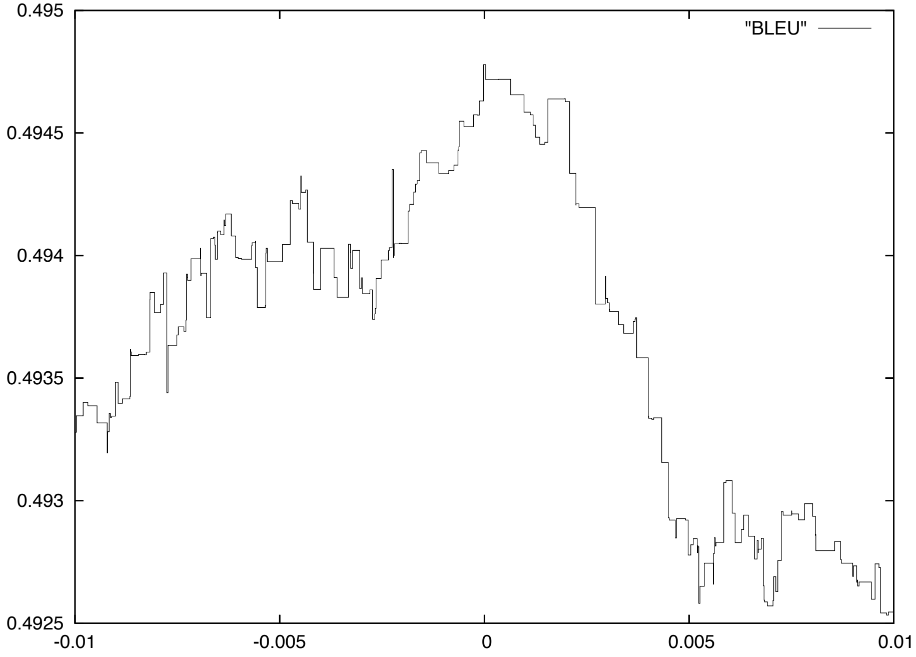
Och's minimum error rate training (MERT)

- **Line search** for best feature weights

```
given:  sentences with n-best list of translations
iterate n times
    randomize starting feature weights
        iterate until convergences
            for each feature
                find best feature weight
                update if different from current
return best feature weights found in any iteration
```

BLEU error surface

- Varying one parameter: a ragged line with many local optima



Unstable outcomes: scores vary

- Even different scores with different runs (varying 0.40 on dev, 0.89 on test)

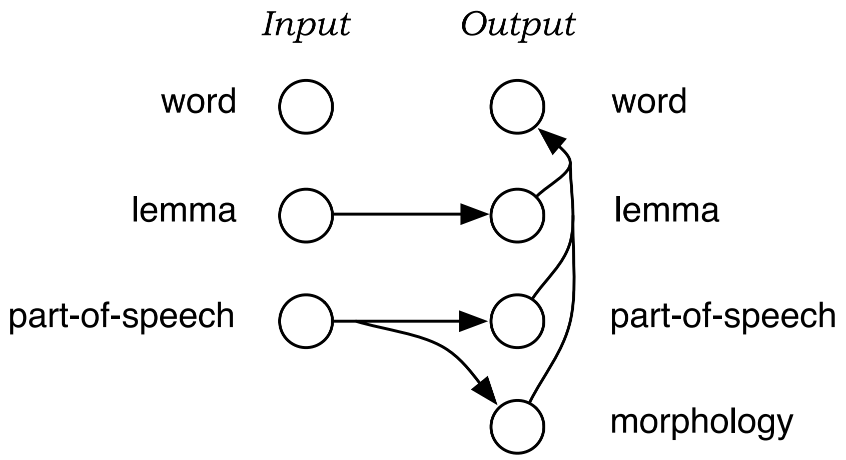
run	iterations	dev score	test score
1	8	50.16	51.99
2	9	50.26	51.78
3	8	50.13	51.59
4	12	50.10	51.20
5	10	50.16	51.43
6	11	50.02	51.66
7	10	50.25	51.10
8	11	50.21	51.32
9	10	50.42	51.79

More features: more components

- We would like to add **more components** to our model
  - multiple language models
  - domain adaptation features
  - various special handling features
  - using linguistic information

→ MERT becomes even **less reliable**
  - runs many more iterations
  - fails more frequently

More features: factored models



- Factored translation models break up phrase mapping into smaller steps
  - multiple translation tables
  - multiple generation tables
  - multiple language models and sequence models on factors

→ **Many more features**

Millions of features

- Why **mix** of discriminative training and generative models?
- Discriminative training of all components
  - phrase table [Liang et al., 2006]
  - language model [Roark et al, 2004]
  - additional features
- **Large-scale** discriminative training
  - millions of features
  - training of full training set, not just a small development corpus

Model

SMT as a **structured prediction** task.

- Local score :

$$s(\mathbf{f_i}, \mathbf{e_i}) = \mathbf{w} \cdot \mathbf{\Phi}(\mathbf{f_i}, \mathbf{e_i})$$

- Translation score :

$$\begin{aligned} s(\mathbf{f}, \mathbf{e}) &= \sum_{(f_i, e_i) \in \mathbf{e}} s(\mathbf{f_i}, \mathbf{e_i}) \\ &= \sum_{(f_i, e_i) \in \mathbf{e}} \mathbf{w} \cdot \mathbf{\Phi}(\mathbf{f_i}, \mathbf{e_i}) \end{aligned}$$

- Decoding :

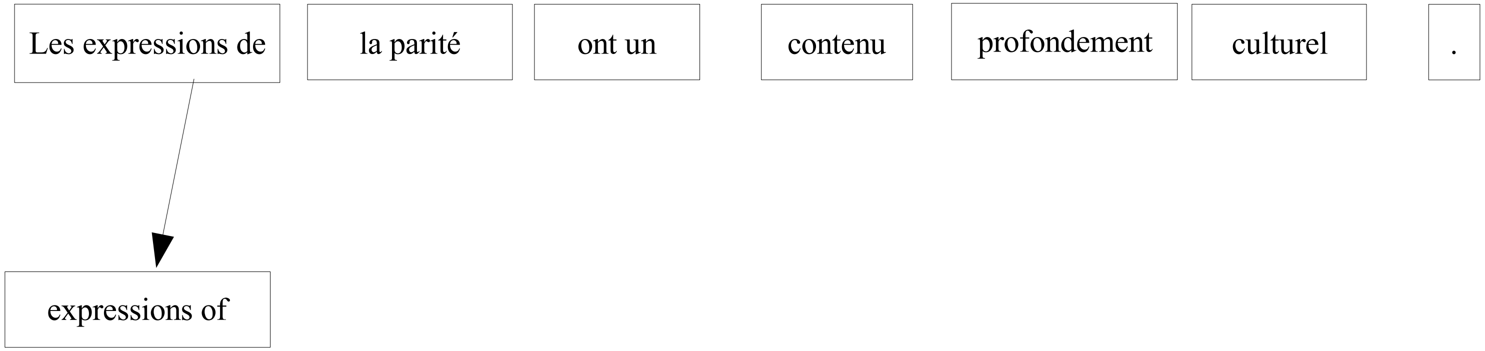
$$\hat{e} = \mathrm{argmax}_{\mathbf{e}} s(\mathbf{f}, \mathbf{e})$$

Featured representation

$$s(\mathbf{f_i}, \mathbf{e_i}) = \mathbf{w} \cdot \mathbf{\Phi}(\mathbf{f_i}, \mathbf{e_i})$$

- Φ: multidimensional feature vector representation
- Can throw in arbitrary features in the model
  - Model can learn from negative evidence e.g downweight *“the the”*
  - Complex interactions between features

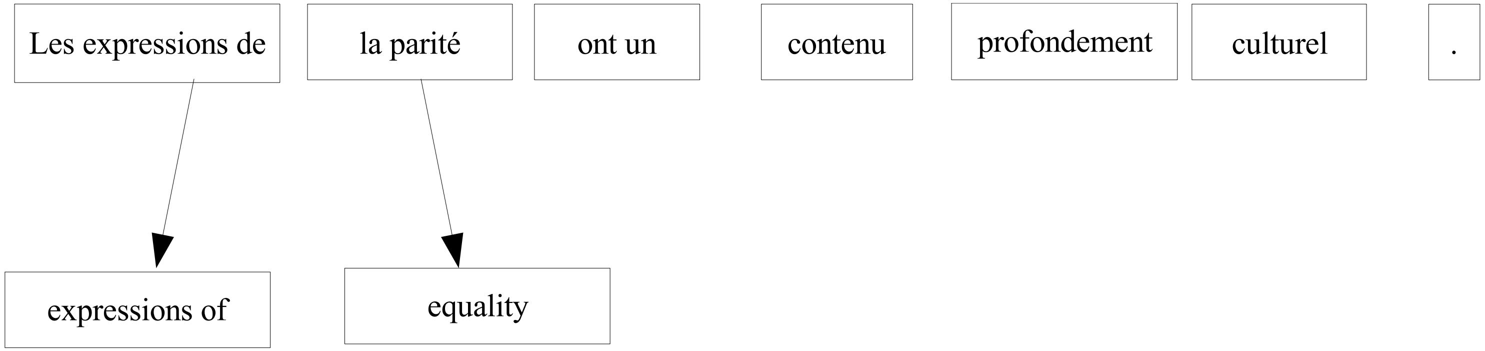
Example



$$\Phi_{100}(\mathbf{f},\mathbf{e}) = \begin{cases} 1 & \text{if } \mathbf{f_i} = \text{“les expressions de”} \wedge\ \mathbf{e_i} = \text{“expressions of”} \\ 0 & \text{otherwise} \end{cases}$$

$$\Phi_{241}(\mathbf{f},\mathbf{e}) = \begin{cases} 1 & \text{if } \mathbf{distortion = 0} \wedge\ \mathbf{f_{i-1}} = \text{“START”} \wedge\ \mathbf{f_i} = \text{“les expressions de”} \\ 0 & \text{otherwise} \end{cases}$$

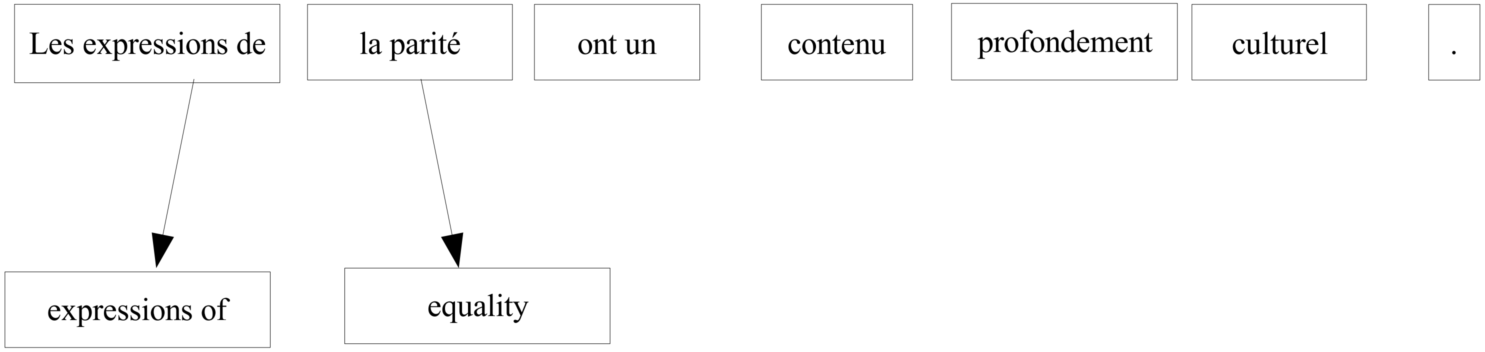
Example



$$\Phi_{729}(\mathbf{f},\mathbf{e}) = \begin{cases} 1 & \text{if } \mathbf{last2TgtWords} = \text{“of equality”} \\ 0 & \text{otherwise} \end{cases}$$

$$\Phi_{730}(\mathbf{f},\mathbf{e}) = \begin{cases} \\ 1 & \text{if } \mathbf{last3TgtWords} = \text{“expressions of equality”} \\ 0 & \text{otherwise} \end{cases}$$

Example



$$\Phi_{317}(\mathbf{f},\mathbf{e}) = \begin{cases} 1 & \text{if } \textbf{orientation} = \text{“MONO”} \wedge\ \mathbf{f_{i-1}} = \text{“les expressions de”} \\ & \wedge\ \mathbf{f_i} = \text{“parite”} \wedge\ \mathbf{e_{i-1}} = \text{“expressions of”} \\ & \wedge\ \mathbf{e_i} = \text{“equality”} \\ 0 & \text{otherwise} \end{cases}$$

Training regimes

$$s(\mathbf{f}, \mathbf{e}) \quad = \quad \sum_{(f_i, e_i) \in \mathbf{e}} \mathbf{w} \cdot \mathbf{\Phi}(\mathbf{f_i}, \mathbf{e_i})$$

- Supervised training : given training set $\mathbf{T} = \{(\mathbf{f_t}, \mathbf{e_t})\}_{\mathbf{t=1}}^{\mathbf{T}}$, estimate **w**
  - Likelihood based models:
    - $*$ Expectations of features across the structure
  - Margin-based methods:
    - $*$ n-best or marginal distribution across graphical structure
    - $*$ Perceptron [Collins, 2002]: only need argmax computation
    - $*$ Approximate large margin: MIRA [Crammer and Singer, 2003]

Perceptron

Requirements:

- Training data: $\mathbf{T} = \{(\mathbf{f_t}, \mathbf{e_t})\}_{\mathbf{t=1}}^{\mathbf{T}}$
- $\hat{\mathbf{e}} = \text{argmax}_{\mathbf{e}} s(\mathbf{f}, \mathbf{e})$
  - Exact computation intractable → beam search
- $\Phi(\mathbf{f_t}, \hat{\mathbf{e}})$
- $\Phi(\mathbf{f_t}, \mathbf{e_t})$

Update rule: $\mathbf{w^{(i+1)}} = \mathbf{w^i} + \mathbf{\Phi}(\mathbf{f_t}, \mathbf{e_t}) - \mathbf{\Phi}(\mathbf{f_t}, \hat{\mathbf{e}})$

Intuition:

- Boost features in correct output and penalise features in incorrect prediction

MIRA

Requirements:

- **T**, $\hat{\mathrm{e}}$, $\Phi(\mathbf{f_t}, \hat{\mathbf{e}})$, $\Phi(\mathbf{f_t}, \mathbf{e_t})$
- Loss function, $\mathbf{L}(\mathbf{e_t}, \hat{\mathrm{e}}) \rightarrow$ measures goodness of prediction wrt to gold standard

Updates weighted by **loss** :

$$\begin{aligned} min & \quad ||\mathbf{w_{i+1}} - \mathbf{w_i}|| \\ s.t & \quad s(\mathbf{f_t}, \mathbf{e_t}) - s(\mathbf{f_t}, \hat{\mathbf{e}}) \geq \mathbf{L}(\mathbf{e_t}, \hat{\mathbf{e}}) \\ \forall \hat{\mathbf{e}} & \quad \in best_k(\mathbf{f_t}; \mathbf{w^{(i)}}) \end{aligned}$$

Problem: overfitting

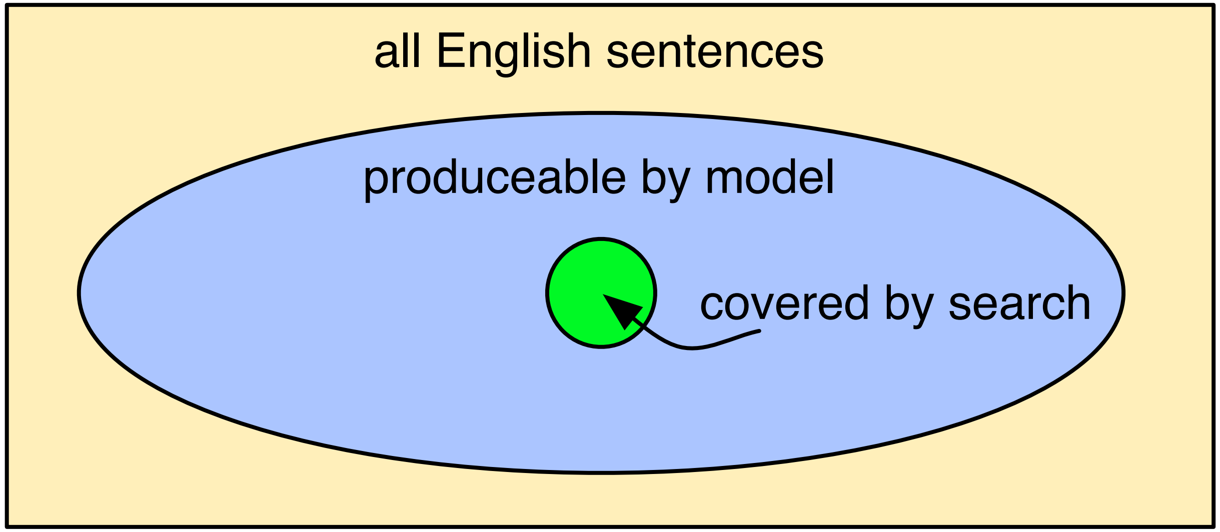
- Fundamental problem in machine learning
  - what works best for training data, may not work well in general
  - **rare, unrepresentative features** may get too much weight
- **Especially severe problem** in phrase-based models
  - **long phrase pairs** explain well *individual sentences*
  - ... but are less general, *suspect to noise*
  - EM training of phrase models [Marcu and Wong, 2002] has same problem

Solutions

- **Restrict to short phrases**, e.g., maximum 3 words (current approach)
  - limits the power of phrase-based models
  - ... but not very much [Koehn et al, 2003]
- **Restrict to short features** : window of 3 words
- **Jackknife**
  - collect phrase pairs from one part of corpus
  - optimize their feature weights on another part
- IBM direct model: **only one-to-many** phrases [Ittycheriah and Salim Roukos, 2007]

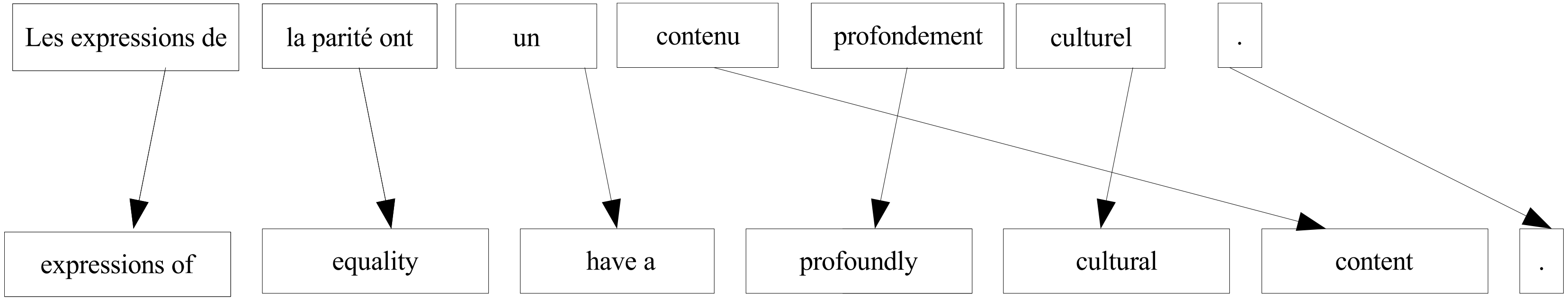
Problem: reference translation

- Supervised training assumes knowledge of gold standard, but...
- Reference translation may not be produceable by model



Problem: reference translation

- If produceable by model → we can compute feature scores
- If not → we can not
- Matching reference string not enough, we want to learn from good phrasal alignments too.



- Multiple ways of going from source to target (if reachable). Is there a **reference phrasal alignment** ?
- Let's just ignore alignments for now...

Update strategies

- **Skip sentences**, for which reference can not be produced
  - invalidates large amounts of training data, biases model to shorter sentences
- Declare candidate translations closest to reference as **surrogate**
  - closeness measured for instance by smoothed BLEU score
  - may be not a very good translation: odd feature values, training is severely distorted

Update strategies

- **Local update**:
  - When including all sentences: surrogate reference picked from 1000-best list using maximum *smoothed BLEU score* with respect to reference translation.
  - **Dynamic reranking**.
- **Min Loss update**:
  - Modify regular decoder to use smoothed BLEU as scoring function.
  - Store min loss candidate for each training instance.

Experiments

Czech-English task - Prague Dependecy treebank, 21K training sentences. **Only binary features**

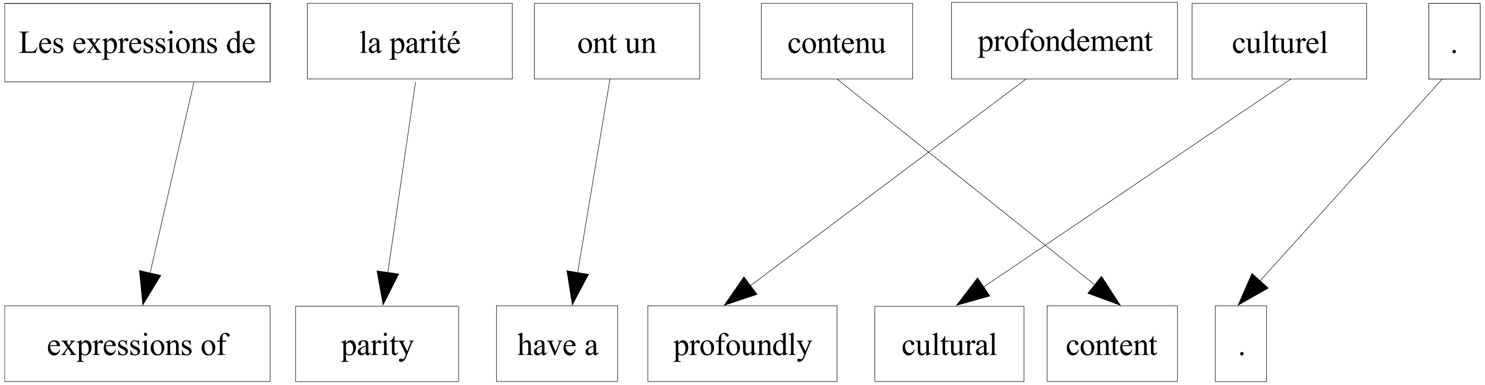
- phrase table features
- lexicalized reordering features
- distortion features
- source and target phrase ngram

Results

Training scheme	BLEU	Length ratio
Pharaoh - MERT	34.53	0.978
Perceptron - local	28.09	0.906
1-best MIRA - local	27.64	0.911
Perceptron - min loss	24.04	0.881
1-best MIRA - min loss	25.24	0.881

Discussion

- Min Loss performing much worse than local updates - why ?
- Local updates more conservative than min loss update
- Loss function ignores **alignments**
- Can produce “good” translations using “dodgy” alignments.
- Loss function insensitive to **paraphrasing**



- Short output - model bias ?

Summary

- Discriminative models allow us to incorporate lots of features
- Proposed model = millions of features (phrase pair, ngram, lexicalised reordering)
- Train on whole corpus
- Margin based learning algorithms
- Problems:
  - Discriminative training: Requires featured representation of gold standard
  - Featured representation of gold standard not always available
  - Model biased towards short output

Future work

- What is a good reference? Paraphrasing to extend reference set.
- Loss functions - sensitive to alignments, lexical choices etc
- mix of binary and real-valued features
- scaling up

More and more features are unavoidable, let's deal with them