Phrase-Based & Neural Unsupervised Machine Translation

Author: Lample et. al

Presenter: Michael Przystupa

Outline

- ~~Abstract~~
- Introduction
- Principles of Unsupervised MT
- Unsupervised MT systems
- Experiments
- ~~Related Work~~
- Conclusions & Future work

Introduction

- Lots of language pairs with limited labeled data
    - e.g. English - Romanian, English - Urdu
- Can separate monolingual text corpuses be used to train a machine translation system?
    - Lots of work done in semi-supervised setting
    - Unsupervised setting less studied
        - Lample et. al. 2018, Artetxe et. al. 2018

Introduction: Contributions

1. Identify general principles of unsupervised MT
2. Apply these principles to train several systems
    a. Neural Network
    b. Phrase-based Statistical MT
3. Show improved BLEU scores on benchmarks with models
4. Demonstrate SOTA performance on low-resource language pairs

Principles of Unsupervised MT

1. Careful initialization of the MT system
2. Leverage Language Model
3. Iterative Back-translation
4. Constrain Latent Representation
    a. Neural network specific
    b. not technically part of principles

Principles: Initialization

- You need a dictionary of at least word level translation
- Options:
  a. Use an existing bilingual dictionary (Klementiev et. al. 2012)
  b. Infer a dictionary from monolingual data (Artetxe et. al. 2017, Conneau et. al. 2018)
- Conneau et al 2018 (briefly):
  - Current best approach for learning Unsupervised dictionary
  - Adversarial training with distance metrics
  - “Broad” idea in picture
  - Available in MUSE project

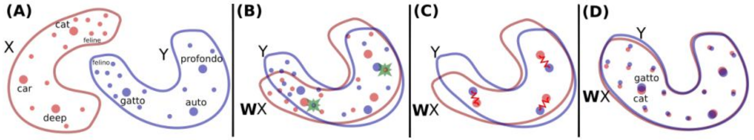
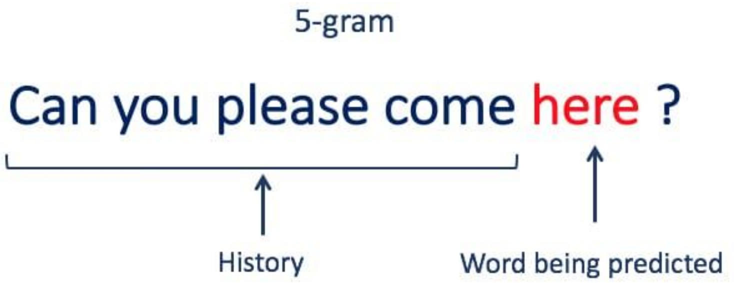


Figure 1: Toy illustration of the method. **(A)** There are two distributions of word embeddings, English words in red denoted by X and Italian words in blue denoted by Y, which we want to align/translate. Each dot represents a word in that space. The size of the dot is proportional to the frequency of the words in the training corpus of that language. **(B)** Using adversarial learning, we learn a rotation matrix W which roughly aligns the two distributions. The green stars are randomly selected words that are fed to the discriminator to determine whether the two word embeddings come from the same distribution. **(C)** The mapping W is further refined via Procrustes. This method uses frequent words aligned by the previous step as anchor points, and minimizes an energy function that corresponds to a spring system between anchor points. The refined mapping is then used to map all words in the dictionary. **(D)** Finally, we translate by using the mapping W and a distance metric, dubbed CSLS, that expands the space where there is high density of points (like the area around the word “cat”), so that “hubs” (like the word “cat”) become less close to other word vectors than they would otherwise (compare to the same region in panel (A)).

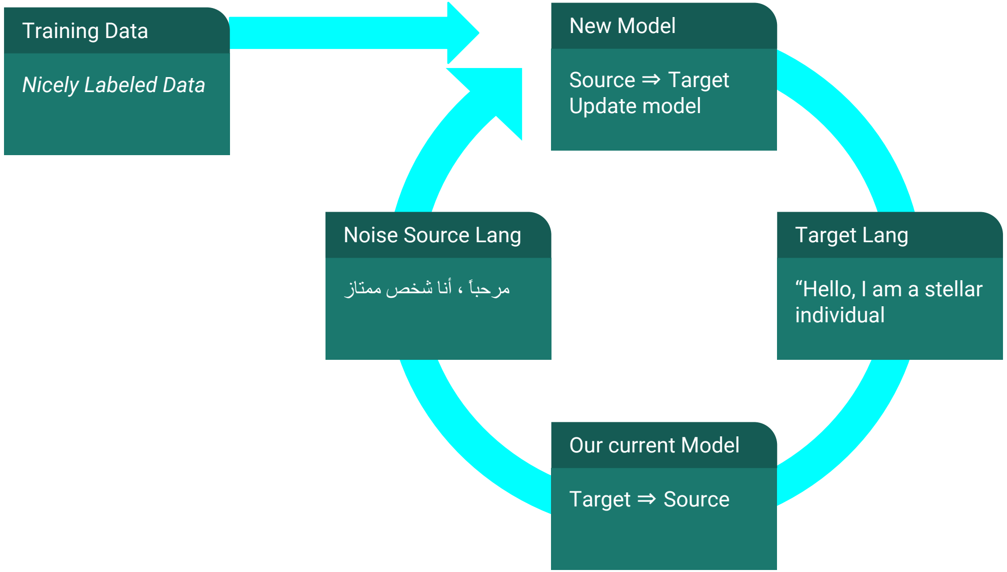
Principles: Language Modeling

- You have tons of monolingual data, so building a good L.M. should be easy
- Defines a prior over the language to give some expectations
    - Can use to see how good translations align with sequences we'd expect to see
    - Can use to do local word substitutions or word swapping



Principles: Iterative Back-translation

- Generate noisy source language training data from target language
- *Sennrich et al 2015*
  - *Improving Neural Machine Translation Models with Monolingual Data*



Training Data

Nicely Labeled Data

New Model

Source ⇒ Target
Update model

Target Lang

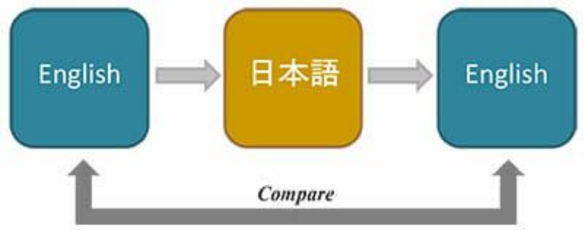
“Hello, I am a stellar individual

Our current Model

Target ⇒ Source

Noise Source Lang

مرحباً ، أنا شخص ممتاز



Principles: Visualization

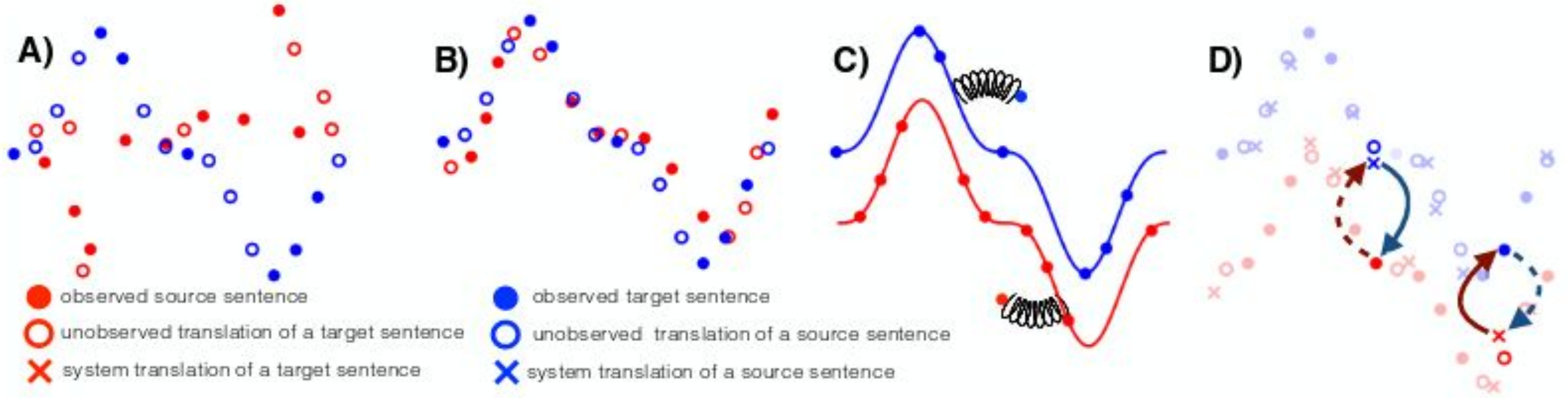
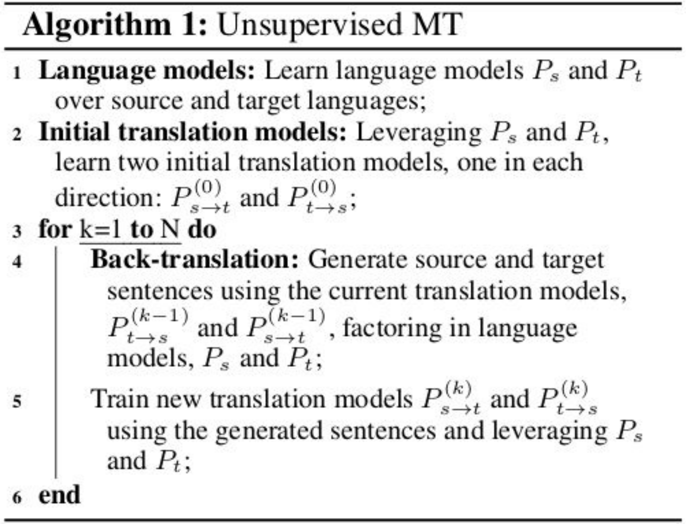

A)
B)
C)
D)
● observed source sentence
○ unobserved translation of a target sentence
✕ system translation of a target sentence
● observed target sentence
○ unobserved translation of a source sentence
✕ system translation of a source sentence

Figure 1: Toy illustration of the three principles of unsupervised MT. **A)** There are two monolingual datasets. Markers correspond to sentences (see legend for details). **B)** First principle: **Initialization**. The two distributions are roughly aligned, e.g. by performing word-by-word translation with an inferred bilingual dictionary. **C)** Second principle: **Language modeling**. A language model is learned independently in each domain to infer the structure in the data (underlying continuous curve); it acts as a data-driven prior to denoise/correct sentences (illustrated by the spring pulling a sentence outside the manifold back in). **D)** Third principle: **Back-translation**. Starting from an observed source sentence (filled red circle) we use the current source → target model to translate (dashed arrow), yielding a potentially incorrect translation (blue cross near the empty circle). Starting from this (back) translation, we use the target → source model (continuous arrow) to reconstruct the sentence in the original language. The discrepancy between the reconstruction and the initial sentence provides error signal to train the target → source model parameters. The same procedure is applied in the opposite direction to train the source → target model.

Unsupervised MT systems

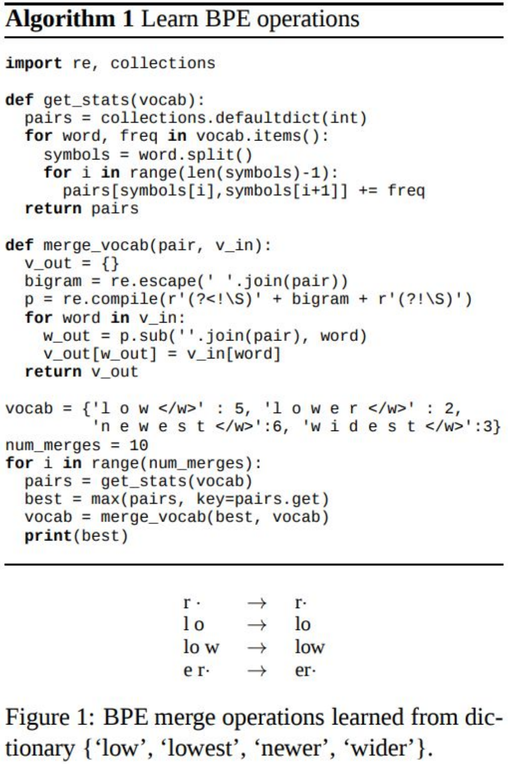
- Built 2 systems:
  - Neural Machine Translation
  - PBSMT
- Outline roles of principles for each model
- Notation:
  - S: Source Sentences
  - T: Target Sentences
  - $P_{S \rightarrow T}$: Language model from S → T
  - $P_{T \rightarrow S}$: Language model from T → S



```
Algorithm 1: Unsupervised MT
Language models: Learn language models $P_s$ and $P_t$ over source and target languages;
Initial translation models: Leveraging $P_s$ and $P_t$, learn two initial translation models, one in each direction: $P^{(0)}_{s \rightarrow t}$ and $P^{(0)}_{t \rightarrow s}$;
for k=1 to N do
    Back-translation: Generate source and target sentences using the current translation models, $P^{(k-1)}_{t \rightarrow s}$ and $P^{(k-1)}_{s \rightarrow t}$, factoring in language models, $P_s$ and $P_t$;
    Train new translation models $P^{(k)}_{s \rightarrow t}$ and $P^{(k)}_{t \rightarrow s}$ using the generated sentences and leveraging $P_s$ and $P_t$;
end
```

Unsupervised NMT: Initialization

- Initialization Process:
    1. Combine monolingual datasets
    2. Apply Byte-pair encoding to joint data set
        - Similar languages will have similar BPE encodings
            - Otherwise we do the Conneau et. al. 2018 method
    3. Learn Token Embeddings on BPE encodings
        - Means there are no unknown words
- Byte-pair Encoding:
    - Seinnrich 2016
        - *Neural Machine Translation of Rare*…
    - Merge frequent character pairs into single token
    - Num Embeddings = #init characters + #Merge operations



Algorithm 1 Learn BPE operations

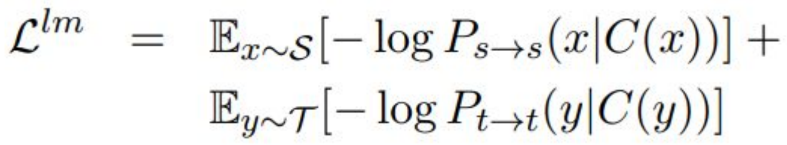
```
import re, collections

def get_stats(vocab):
  pairs = collections.defaultdict(int)
  for word, freq in vocab.items():
    symbols = word.split()
    for i in range(len(symbols)-1):
      pairs[symbols[i],symbols[i+1]] += freq
  return pairs

def merge_vocab(pair, v_in):
  v_out = {}
  bigram = re.escape(' '.join(pair))
  p = re.compile(r'(?<!\S)' + bigram + r'(?!\S)')
  for word in v_in:
    w_out = p.sub(''.join(pair), word)
    v_out[w_out] = v_in[word]
  return v_out

vocab = {'l o w </w>' : 5, 'l o w e r </w>' : 2,
         'n e w e s t </w>':6, 'w i d e s t </w>':3}
num_merges = 10
for i in range(num_merges):
  pairs = get_stats(vocab)
  best = max(pairs, key=pairs.get)
  vocab = merge_vocab(best, vocab)
  print(best)
```

r ·	→	r·
l o	→	lo
lo w	→	low
e r·	→	er·

Figure 1: BPE merge operations learned from dictionary {'low', 'lowest', 'newer', 'wider'}.

Unsupervised NMT: Language Modeling

- Optimize denoising autoencoder loss function
    - *Refer to image*
    - $P_{s\text{-->}T}$ & $P_{T\text{-->}S}$ composition of decoder & encoder on S & T
- C is a noise model from Lample et. al. 2018
    1. Drop words with probability p_{wd}
    2. Slight word shuffle
        - Permute operation **σ** on sequence
        - Condition ∀i in {1, n}, |**σ** (i) - i| ≤ k
            - i : word (maybe a word embedding?)
            - n : length of sequence
            - k : hyperparameter to of how noisy to make it

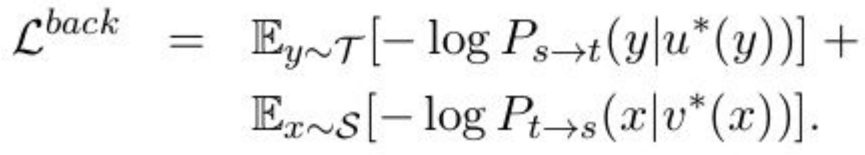


$$\begin{aligned}\mathcal{L}^{lm} &= \mathbb{E}_{x\sim\mathcal{S}}[-\log P_{s\to s}(x|C(x))] + \\ &\quad \mathbb{E}_{y\sim\mathcal{T}}[-\log P_{t\to t}(y|C(y))]\end{aligned}$$

- Definition of **σ**
    - Generate random sequence q , q_i = i + U(0, α) , α: hyperparameter (k + 1 in paper)
    - **σ** sorts sequence q and accepts based on condition
    - **WARNING:** biased samples

Unsupervised NMT: Back-translation

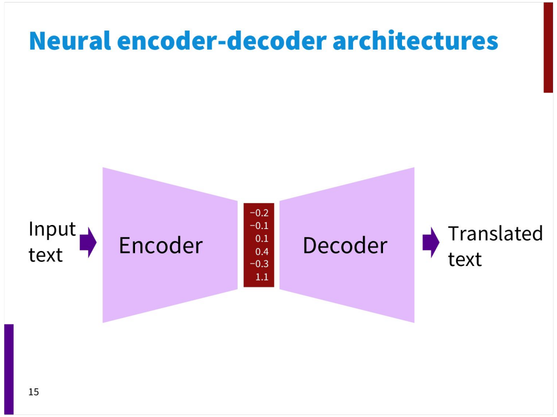
- u*(y) = argmax $P_{S\rightarrow\ T}$(u | y)
- v*(x) = argmax $P_{T\text{-->}S}$(v | x)
- End up with noisy translation pairs (u*(y), y) & (x, v*(x))
    - Input u* & v* as inputs to see if can recreate the “labels” that generated u* & v*
    - Use to minimize loss function (refer to image)
    - Do not backprop errors on the generated u* or v*
- Final loss objective: L^{back} + L^{lm}



$$\begin{aligned}\mathcal{L}^{back} \quad = \quad & \mathbb{E}_{y\sim\mathcal{T}}[-\log P_{s\to t}(y|u^*(y))] + \\ & \mathbb{E}_{x\sim\mathcal{S}}[-\log P_{t\to s}(x|v^*(x))].\end{aligned}$$

Unsupervised NMT: Sharing Latent Representations

- Interlingua
  - *An artificial international language formed of elements common to the Romance languages, designed primarily for scientific and technical use*
- Remember that we learned embeddings in a joint space between languages sub-words
- Let’s share the encoder for $P_{s \rightarrow T}$ & $P_{T \rightarrow S}$
  - MUST BE DONE
  - Optional: Share decoder as well
    - Add token to specify language decoding



Neural encoder-decoder architectures

Unsupervised PBSMT: Overview

- Koehn et. al. 2003
- Works great in low-resource setting
- For translating x → y: $\text{argmax}_y P(y \mid x) = \text{argmax}_y \; P(x \mid y) \; P(y)$
  - $P(x \mid y)$: phrase table which we need to learn
    - Hard part
    - Going to learn some sort of alignment between bitexts corpus
  - $P(y)$: score provided by language model
    - Easy to do

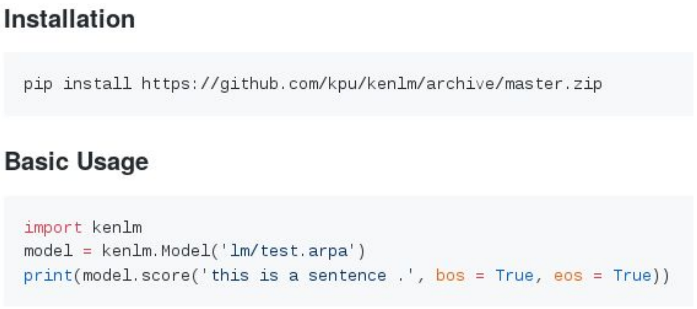
Unsupervised PBSMT: Initialization

- Populate initial phrase table with Conneau et. al. 2018 method
  - Score operation in picture
- Definitions of terms:
  - s_i : i^{th} word in source vocab
  - t_j : j^{th} word in target vocab
  - T : temperature to define peakiness
  - W : rotation matrix
    - Learned with Conneau et. al. 2018 approach
  - e(x) : embedding of x
    - x could be words, phrase, or BPE representation

$$p(t_j|s_i) = \frac{e^{\frac{1}{T}\cos(e(t_j), We(s_i))}}{\sum_k e^{\frac{1}{T}\cos(e(t_k), We(s_i))}},$$

Unsupervised PBSMT: Language Modeling

- KenLM
  - Heafield 2011
  - Optimized language modeling API
  - Fix during training (kinda)
- Alternatively a neural network could be used
- Python Snippet:



Installation

```
pip install https://github.com/kpu/kenlm/archive/master.zip
```

Basic Usage

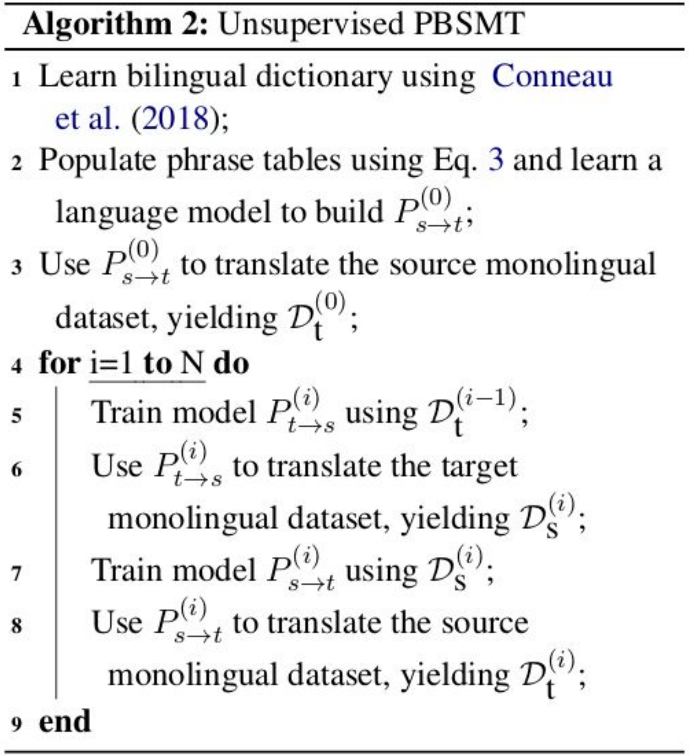
```
import kenlm
model = kenlm.Model('lm/test.arpa')
print(model.score('this is a sentence .', bos = True, eos = True))
```

$$p(w_n|w_1^{n-1}) = p(w_n|w_f^{n-1}) \prod_{i=1}^{f-1} b(w_i^{n-1}).$$

Example LM from Heafield 2011

Unsupervised PBSMT: Iterative Back-Translation

- *Refer to pseudo-code*
  - lines 3 - 8 are back-translation step
  - D_x^i: Generated synthetic translation
- Rational why this works:
  - The phrase table is going to be noise
  - Language model critical to improvement
    - A few fixes here and there & things improve



Algorithm 2: Unsupervised PBSMT

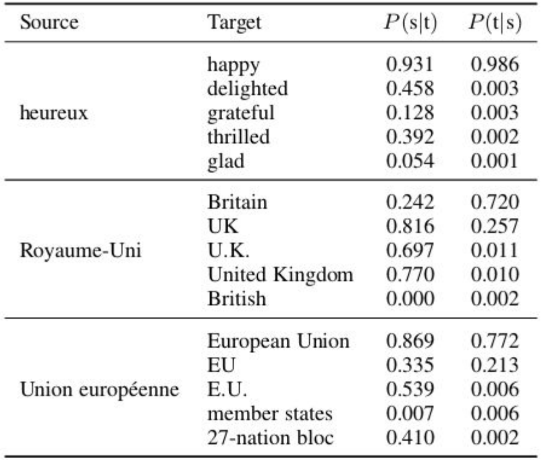
```
1 Learn bilingual dictionary using Conneau et al. (2018);
2 Populate phrase tables using Eq. 3 and learn a language model to build $P^{(0)}_{s \to t}$;
3 Use $P^{(0)}_{s \to t}$ to translate the source monolingual dataset, yielding $\mathcal{D}^{(0)}_{\mathrm{t}}$;
4 for i=1 to N do
5     Train model $P^{(i)}_{t \to s}$ using $\mathcal{D}^{(i-1)}_{\mathrm{t}}$;
6     Use $P^{(i)}_{t \to s}$ to translate the target monolingual dataset, yielding $\mathcal{D}^{(i)}_{\mathrm{s}}$;
7     Train model $P^{(i)}_{s \to t}$ using $\mathcal{D}^{(i)}_{\mathrm{s}}$;
8     Use $P^{(i)}_{s \to t}$ to translate the source monolingual dataset, yielding $\mathcal{D}^{(i)}_{\mathrm{t}}$;
9 end
```

Experiments: Datasets and Methodology

- Language Pairs: English - <French, German, Romanian, Russian, Urdu>
- Data sources:
  - English, German & Russian: WMT News Crawl 2007 - 2017
  - Romanian: WMT News Crawl + WMT’ 2016 (challenge?)
  - Urdu: Jawaird et. et al 2014 + LDC2010T23 + LDC2010T21
- Tokenize data with Moses
- NMT: 60,000 BPE codes
- PBSMT: train w/ True-casing + no diacritics in Romanian

Experiments: Initialization

- NMT
  - Generate embeddings with fastText (Bojanowski et. al. 2017)
    - Learned on joint BPE’s of S & T
- PBSMT
  - Generate embeddings with fastText (Bojanowski et. al. 2017)
    - Learned embeddings on separate n-grams of S & T
  - Too many phrases, so clamp to 300K
    - w/ 200 nearest neighbors in target
  - Bi-grams work slightly better
    - 1 whole BLEU point



Source	Target	$P(\mathrm{s}\vert\mathrm{t})$	$P(\mathrm{t}\vert\mathrm{s})$
heureux	happy	0.931	0.986
	delighted	0.458	0.003
	grateful	0.128	0.003
	thrilled	0.392	0.002
	glad	0.054	0.001
Royaume-Uni	Britain	0.242	0.720
	UK	0.816	0.257
	U.K.	0.697	0.011
	United Kingdom	0.770	0.010
	British	0.000	0.002
Union européenne	European Union	0.869	0.772
	EU	0.335	0.213
	E.U.	0.539	0.006
	member states	0.007	0.006
	27-nation bloc	0.410	0.002

Table 1: **Unsupervised phrase table.** Example of candidate French to English phrase translations, along with their corresponding conditional likelihoods.

Experiments: Training

- NMT
    - Try LSTM & Transformer cells
    - Weight tying: Between encoder & decoder + source & target lang
        - Press and Wolf 2016
    - Embeddings + hidden unit dimensions: 512
    - Adam optimizer
- PBSMT
    - Use Moses API
    - Use Algorithm 2 (we saw this earlier)
    - Translate 5 million sentences each iteration
    - Use phrases up to length 4

MOSES

statystical
machine translation
system

Experiments: Model Selection

- PBSMT
    - Didn’t work really well, mostly used default configs in Moses
- NMT
    - Transformer: Round Robin Bleu validation criterion (S → T → S → T)
    - LSTM: Used validation set of 100 samples
        - Empirically other metric didn’t work as well here

Experiments: Results

- Approach’s worked better than previous work
  - Even untrained PBSMT better
- Compared approach to Supervised Approach
  - Better up until 100,000 region
  - Consistent with low-resource languages results
    - Romanian - English
    - Urdu -English

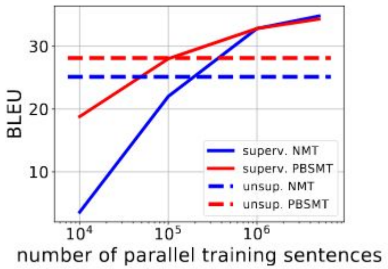
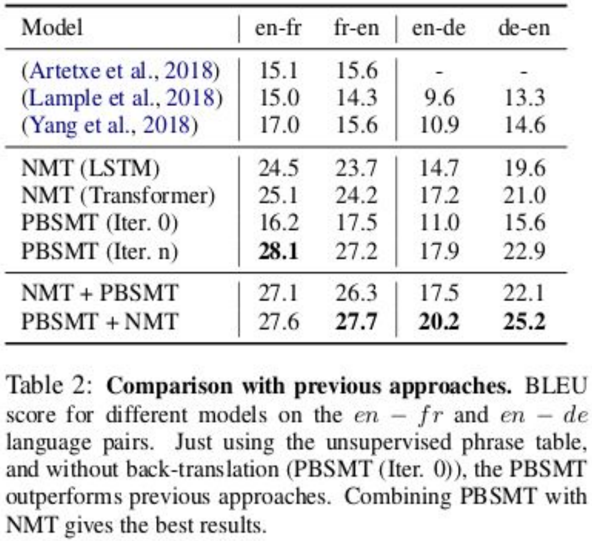


Figure 2: Comparison between supervised and unsupervised approaches on WMT’14 En-Fr, as we vary the number of parallel sentences for the supervised methods.

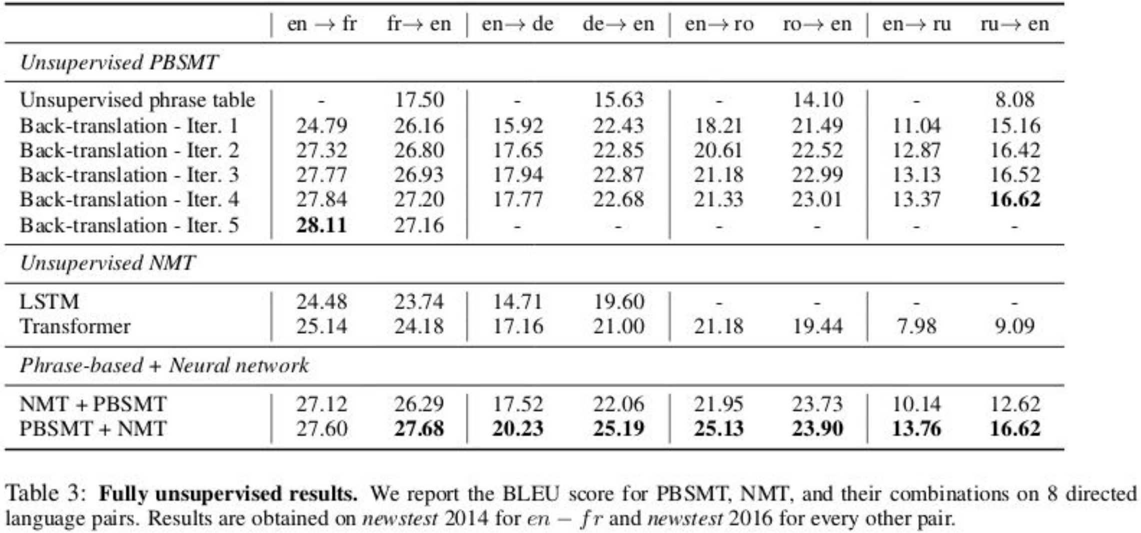


Model	en-fr	fr-en	en-de	de-en
(Artetxe et al., 2018)	15.1	15.6	-	-
(Lample et al., 2018)	15.0	14.3	9.6	13.3
(Yang et al., 2018)	17.0	15.6	10.9	14.6
NMT (LSTM)	24.5	23.7	14.7	19.6
NMT (Transformer)	25.1	24.2	17.2	21.0
PBSMT (Iter. 0)	16.2	17.5	11.0	15.6
PBSMT (Iter. n)	**28.1**	27.2	17.9	22.9
NMT + PBSMT	27.1	26.3	17.5	22.1
PBSMT + NMT	27.6	**27.7**	**20.2**	**25.2**

Table 2: **Comparison with previous approaches.** BLEU score for different models on the $en - fr$ and $en - de$ language pairs. Just using the unsupervised phrase table, and without back-translation (PBSMT (Iter. 0)), the PBSMT outperforms previous approaches. Combining PBSMT with NMT gives the best results.

Experiments: Results

- Back-translation helps
- NMT + PBSMT
  - Use data from NMT in PBSMT
  - Not quite helpful
- PBSMT + NMT
  - Use data from PBSMT in NMT
  - Helped a little



	en → fr	fr→ en	en→ de	de→ en	en→ ro	ro→ en	en→ ru	ru→ en
Unsupervised PBSMT								
Unsupervised phrase table	-	17.50	-	15.63	-	14.10	-	8.08
Back-translation - Iter. 1	24.79	26.16	15.92	22.43	18.21	21.49	11.04	15.16
Back-translation - Iter. 2	27.32	26.80	17.65	22.85	20.61	22.52	12.87	16.42
Back-translation - Iter. 3	27.77	26.93	17.94	22.87	21.18	22.99	13.13	16.52
Back-translation - Iter. 4	27.84	27.20	17.77	22.68	21.33	23.01	13.37	**16.62**
Back-translation - Iter. 5	**28.11**	27.16	-	-	-	-	-	-
Unsupervised NMT								
LSTM	24.48	23.74	14.71	19.60	-	-	-	-
Transformer	25.14	24.18	17.16	21.00	21.18	19.44	7.98	9.09
Phrase-based + Neural network								
NMT + PBSMT	27.12	26.29	17.52	22.06	21.95	23.73	10.14	12.62
PBSMT + NMT	27.60	**27.68**	**20.23**	**25.19**	**25.13**	**23.90**	**13.76**	**16.62**

Table 3: **Fully unsupervised results.** We report the BLEU score for PBSMT, NMT, and their combinations on 8 directed language pairs. Results are obtained on *newstest* 2014 for $en - fr$ and *newstest* 2016 for every other pair.

Experiments: Ablation Study

- Ablation: *The surgical removal of body tissue*
- Punchline: All the principles contribute to good BLEU scores

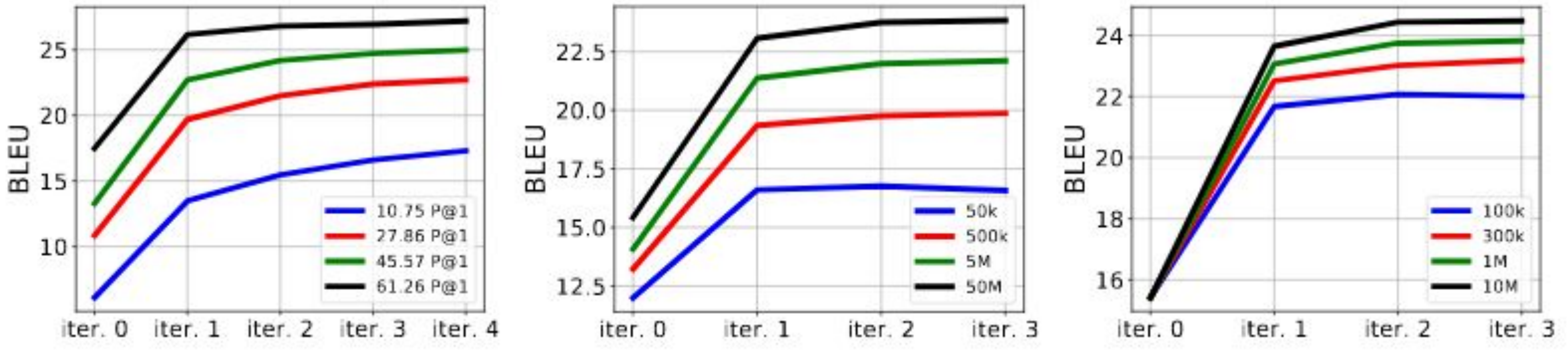
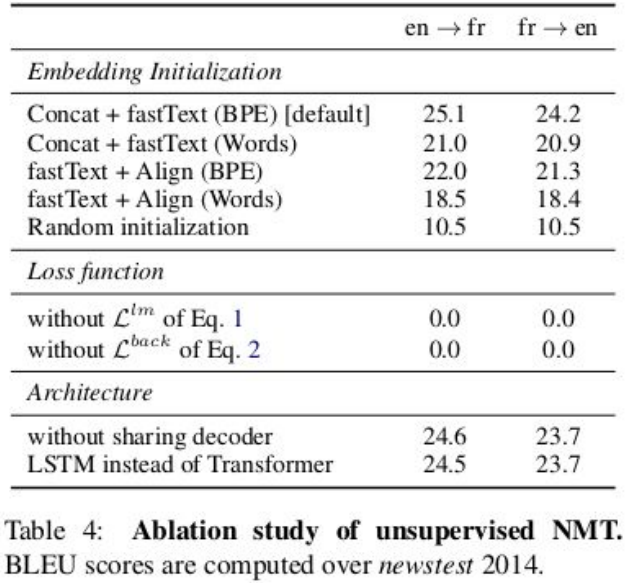


Figure 3: Results with PBSMT on the $fr \rightarrow en$ pair at different iterations. We vary: Left) the quality of the initial alignment between the source and target embeddings (measured in P@1 on the word translation task), Middle) the number of sentences used to train the language models, Right) the number of sentences used for back-translation.

Experiments: Ablation Study

- Ablation: *The surgical removal of body tissue*
- Punchline: All the principles contribute to good BLEU scores



	en → fr	fr → en
Embedding Initialization		
Concat + fastText (BPE) [default]	25.1	24.2
Concat + fastText (Words)	21.0	20.9
fastText + Align (BPE)	22.0	21.3
fastText + Align (Words)	18.5	18.4
Random initialization	10.5	10.5
Loss function		
without $\mathcal{L}^{lm}$ of Eq. 1	0.0	0.0
without $\mathcal{L}^{back}$ of Eq. 2	0.0	0.0
Architecture		
without sharing decoder	24.6	23.7
LSTM instead of Transformer	24.5	23.7

Table 4: **Ablation study of unsupervised NMT.** BLEU scores are computed over *newstest* 2014.

Conclusions

- Here are 3 principles that make unsupervised MT work well
- PBSMT seems to work better than NMT generally
- Open Questions:
    - Are there other important principles as well?
        - Can we apply these principles in a better way?
    - How do we guarantee convergence in iterative process?
    - What about semi-supervised setting?