Assessing Quick Update Methods of Statistical Translation Models

Shachar Mirkin & Nicola Cancedda

IWSLT 2013

transLectures

- Transcription and translation of educational video lectures
- To be integrated into lecture repositories
  - VideoLectures.NET, poliMedia
- 3 transcription languages:
  - English (en), Slovenian (sl), Spanish (es)
- 6 translation language pairs:
  - en $\rightarrow$ {fr, de, sl, es}, sl $\rightarrow$ en, es $\rightarrow$ en
- **ASR** and **SMT**, with **human supervision** for each
- Improving the models
  - By adaptation
  - Based on users' feedback

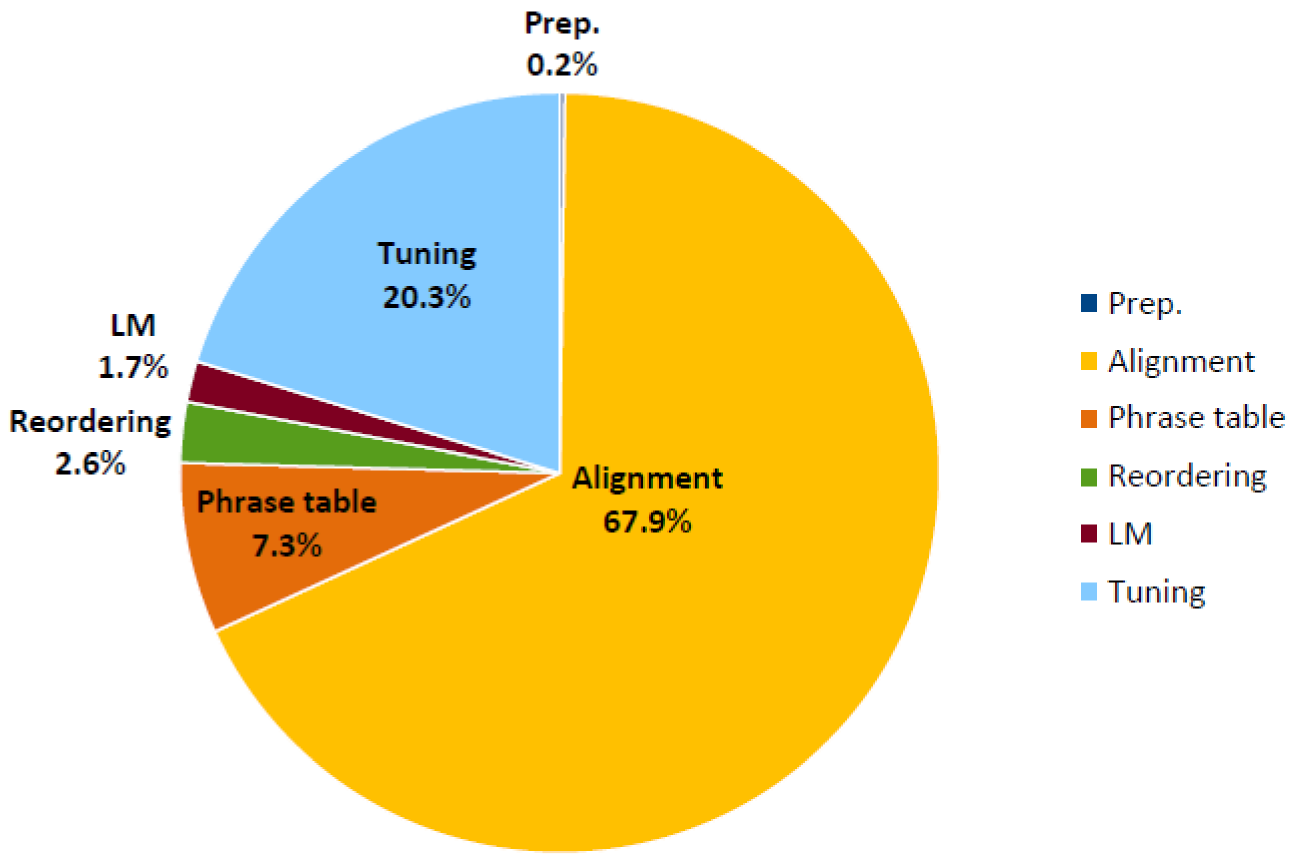
http://www.translectures.eu/

Users' feedback

- Main source of feedback: corrected translation (post-edition)
- Pretty standard procedure
  - When good quality translation is required
- May be tedious and time-consuming
  - We wish to minimize the effort
- Post-edition generates additional training material
  - That we can use to update the model
- We want the process to be *fast*

SMT model updates

SMT model – time per task



- CPU-time
- Exact configuration – in the paper

Incremental training

- Enables updating the model based on new training data
  - Without re-training using all data
    - Much faster

- Using Online EM for alignment instead of “Batch”-EM

- Expected to produce better alignment
  - Vs. aligning only the new data on its own

- Potentially reflecting feedback immediately (**real-time**)

Incremental training for Moses

Moses - Moses/AdvancedFea

www.statmt.org/moses/?n=Moses.AdvancedFeatures

Incremental Training

Introduction

Translation models for Moses are typically batch trained. That is, before training you have all the data you wish to use, you compute the alignments using GIZA, and from that produce a phrase table which you can use in the decoder. If some time later you wish to utilize some new training data, you must repeat the process from the start, and for large data sets, that can take quite some time.

Incremental training provides a way of avoiding having to retrain the model from scratch every time you wish to use some new training data. Instead of producing a phrase table with precalculated scores for all translations, the entire source and target corpora are stored in memory as a suffix array along with their alignments, and translation scores are calculated on the fly. Now, when you have new data, you simply update the word alignments, and append the new sentences to the corpora along with their alignments. Moses provides a means of doing this via XML RPC, so you don't even need to restart the decoder to use the new data.

Note that at the moment the incremental phrase table code is not thread safe.

Initial Training

This section describes how to initially train and use a model which support incremental training.

- Setup the EMS as normal, but use a modified version of GIZA++.
- Add the line:

```
training-options = "-final-alignment-model hmm"
```

to the TRAINING section of your experiment configuration file.

- Train the system using the initial training data as normal.
- Modify the moses.ini file found in `<experiment-dir> /evaluation/filtered.<evaluation-set>.<run-number>` to have a ttable-file entry as follows:

```
8 0 0 3 <path-to-source-corpus> <path-to-target-corpus> <path-to-alignments>
```

Suffix Arrays for SMT

- The entire training data is kept in memory
  - Instead of generating a phrase table in advance
- Scores are computed on the fly

  (Callison-Burch and Bannard, 2005)

- *Dynamic suffix arrays* (Levenberg et al., 2010):
  - Additions and deletions of parallel sentences are also possible
  - Enabling incremental training

Inc GIZA w/ suffix arrays in Moses

- Incremental GIZA (Levenberg et al., 2010)
  - HMM-based alignments and IBM Model 1
  - Stepwise Online EM
    - Updating model in mini-batches
    - Interpolating counts of old and new data

- Results are comparable to GIZA++ when used without updates

- Updating:
  1. Preprocess the new data
  2. Update vocabularies, co-occurrences & alignments
  3. Align the new data
  4. Insert new sentence-pairs into the suffix array via the *Moses Server*

Inc. GIZA w/ SA in Moses – Limitations

- Pros (reminder)
  - No need to re-align the entire data
  - Better statistics for aligning the new data
  - Potentially: real-time update

- Limitations*
  - High memory usage
  - Can't save updated model to disk
  - Reverse translation probability features not computed
  - LM not updated
  - **Insertion takes a long time**
    - The system is unusable in the meanwhile
  - **Results are inferior**
    - Just the missing features? Not sure

* Before new suffix array version of summer 2013

Quick SMT model updates

Update cycles

- **Slow**
  - E.g. weekly
  - Any kind of task
    - Tuning, retraining, binarizing

- **Quick**
  - E.g. daily
  - Only short tasks
    - Preprocessing, training of small corpora

Quick updates - Goal

Quickly update MT models in between slow retraining

- Expecting to:
  - Improve vs. last slow update (“batch”)
  - Get close to next slow update
  - Via a quick method

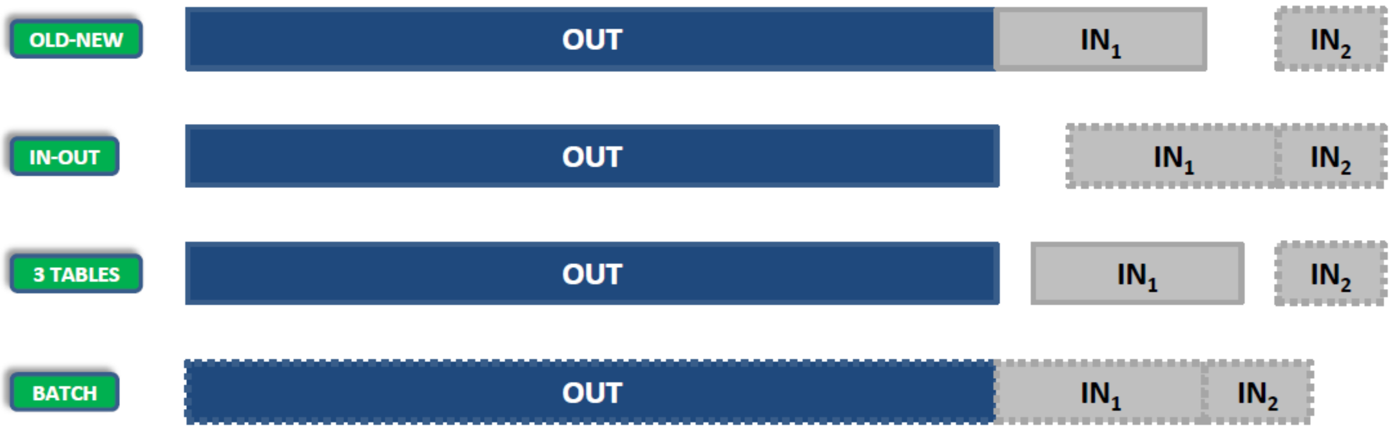
Domain adaptation

- We start with little in-domain data
  - Expecting to obtain more as we proceed
  - Depend also on large out-of-domain data

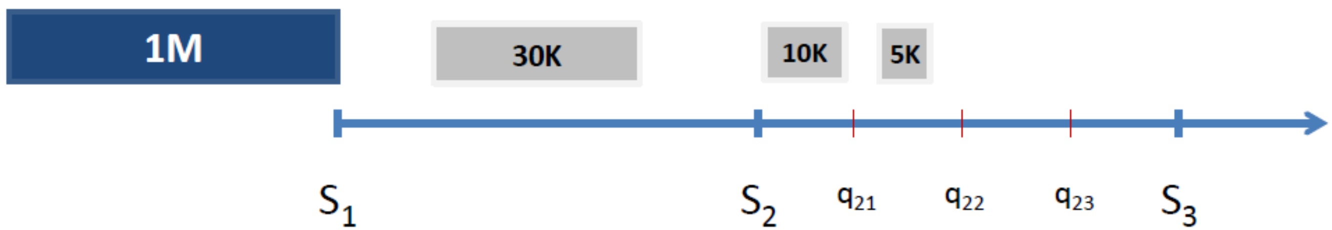
- Prior work
  - Separating in- and out-of-domain data helps
    - Both for phrase tables and language models

Assessed configurations – PT setting

- *OLD-NEW*: 2 phrase tables (PTs): old/new data (relative to last slow update)
  - *Very quick*
- *IN-OUT*: 2 PTs: in/out of domain
  - *Slower; proved to work well for domain adaptation*
- *3-TABLES*: 1 table for out; 2 for in-domain: old & new
  - *Potentially combines benefits of the above two*
- *BATCH:* all **concatenated**



Effort by configurations - example



q_{22}

Config./ Task	Prep.	Alignment	PT	LM
OLD-NEW	5K	15K	15K	15K
IN-OUT	5K	45K	45K	45K
3-TABLES	5K	15K	15K	15K
BATCH	5K	1,045K	1,045K	1,045K

Experiments

Setting

- 2 spoken language datasets:
  - transLectures (TL)
  - TED talks (WIT^3)
- Europarl as out-of-domain

		TL (EN-FR)	WIT^3 (IT-EN)
out-of-domain	**Training**	Europarl 1M	Europarl 1M
In-domain	**Training**	4K	40K
	Development	1K	1K
	Test	1360	1K

- **Phrase-based SMT**: Moses
  - *either* option for multiple phrase tables
- **Language models**: 5-gram SRILM
- **Tuning**: MIRA
- **Evaluation**: Averaged sentence-level BLEU
- **Reordering table** not updated

Slow updates

- Using the complete in-domain data

Dataset	Configuration	BLEU
transLectures	Baseline	23.9
	BATCH, PT only	27.9
	BATCH, complete	28.3
WIT3	Baseline	29.4
	BATCH, PT only	30.9
	BATCH, complete	30.7

- Adding in-domain data significantly improves results
  - (even only 4K sentence-pairs)

Quick PT updates

- Generating only phrase tables with the new data

Dataset	Configuration	BLEU
transLectures	OLD-NEW	29.4
	IN-OUT	29.7
	3-TABLES	30.2
WIT3	OLD-NEW	31.2
	IN-OUT	31.7
	3-TABLES	31.2

vs. 28.3

vs. 30.9

- All are better (and faster) than batch retraining

Quick PT & LM updates

- Updating also the LMs
  - Matching LM configuration to PTs'

Dataset	Configuration	BLEU
transLectures	OLD-NEW	31.2
	IN-OUT	31.8
	3-TABLES	31.6
WIT3	OLD-NEW	32.3
	IN-OUT	33.1
	3-TABLES	32.3

vs. 30.2

vs. 31.7

- Separating also the LMs further helps
  - But not separating them too much

Quick PT & LM updates

Dataset	Configuration	BLEU
transLectures	OLD-NEW	31.2
	IN-OUT	31.8
	3-TABLES	31.6
WIT3	OLD-NEW	32.3
	IN-OUT	33.1
	3-TABLES	32.3

Only LM is (quickly) updated

Configuration	BLEU
Single LM	29.4
Separate LMs	31.4

IN-OUT

- Separating only the LMS helps vs. baseline, but less than PT separation
  - It's not enough to separate only the LM

Separating the LMs for batch training

- Single phrase table, but separate *IN-OUT* LMs
  - Slow update (but not the slowest)

Dataset	Configuration	BLEU
TL	BATCH, single LM	28.3
	BATCH, separate LMs	31.6
WIT3	BATCH, single LM	30.7
	BATCH, separate LMs	32.6

Vs. 31.8

Vs. 33.1

- Separating LMs helps batch training
- Slower & not better than separation of also the PTs

No adaptation

- Only WIT3, starting with 30K and updating with 10K
- IN-OUT and 3-TABLES not relevant

Configuration	BLEU
Baseline	28.2
BATCH	29.2
OLD-NEW	28.5
OLD-NEW, single LM	28.9

- Slow update gets best results
- Out-of-domain data is still useful (33.1)
  - Then - quick updates are really necessary

Other results

- Re-ordering table isn't significant
  - (for the language pairs we tested)
  - Needn't be constantly updated

- Tuning can be deferred to slow update
  - If no significant change of configuration

Practical recommendations

- Phrase tables:
  1. Not much in-domain data: use IN-OUT
  2. A lot of in-domain data: use 3-TABLES'
     - *Most* of older in-domain data in one table
     - Reserve some for the new table, to be trained with the new data

- LMs:
  - Use IN-OUT LMs, even if using 3-TABLES
  - Pretty quick in any case

  – Depends on exact conditions
    - Computing power, amount of data

- Keep an eye of suffix arrays & inc. training in Moses
  - A new version is (being) implemented
  - May be combined with quick updates configurations

Thank you!

(and thanks for the very useful reviews)

Tuning

Setting	Configuration	BLEU
TL, 2K; IN-OUT	Baseline	27.76
	Re-tuned	28.45
	Not re-tuned	28.31
TL, 4K; OLD-NEW	Baseline	28.51
	Re-tuned	29.37
	Not re-tuned	29.19