MT Marathon 2013 (Prague)

Proposed Projects

Feel free to start a new entry or add your comments anywhere, in the text or on side.
Projects can be proposed until the first day of MT Marathon, but announcing them earlier might attract more participants, come better prepared etc.

Overview

- All project boaster slides: (username and password ‘mtm’ for read-only access)

Be prepared to present your final report on Friday, Sept 14.
You will have something around 15 minutes to present your project.

Please commit your final slides in PDF here:
svn co https://svn.ms.mff.cuni.cz/svn/mtmarathon-2013/trunk/projects/final-reports

Your username should be name.surname and your password should be 'mtm???' (or it should be known to you for some years already).

List of Projects

Commonspector

(Ondřej Bojar, Aleš Tamchyna, possibly also Niko Papula)
Please e-mail Ondrej and Ales {bojar,tamchyna}@ufal.mff.cuni.cz to meet and get started.

- Prospector is an extension of Eman, our variant of EMS, that automatically searches through MT experiment setups.
- The goal of the project is to use Prospector for community service:
    - Think of Large Hadron Collider where you submit your experiments and (a thousand years later), the results will appear for you on a web page.
    - Or think of a cruise-control extension that regularly checks the actual performance of Moses releases on a few language pairs.
- Prospective participants: Ondřej, Aleš

QuEst

(Lucia Specia, Niko Papula (?))

Quality estimation for human translation

This project extend the QuEst framework for **machine** translation quality estimation (http://www.quest.dcs.shef.ac.uk/) such that it can predict quality for **human** translations. QuEst is implemented in Java, and it provides a module for feature extraction (with a number of existing features and general templates for new features and resources), and a module for machine learning (a wrapper for scikit-learn - in python). The project will start from a corpus of automatic translations annotated manually with translation errors and their categories (http://terra.cl.uzh.ch/terra-corpus-collection.html -- annotations can be viewed online). The task is to find patterns in this corpus and then to design and implement feature extractors that can capture some of these errors, and - if time allows - experiment with machine learning algorithms to test the effectiveness of such features on human (and machine) translation data. These feature extractors could in principle be used for both sentence- and **word-level** QE.

Document-wide quality estimation

This project will extend the QuEst framework to predict the quality of machine translations with new feature extractors and learning methodologies such that quality indicators relating different segments can be considered. Current work on the topic explores features that are local to the segment being analysed (sentences or short paragraphs). This project will exploit the connection/relationship between the current segment and other segments in the document, in

order to model phenomena such as anaphors, lexical cohesion, etc. We will start with a suggestion of possible features (https://docs.google.com/spreadsheet/ccc?key=0AkPSvLhZ0twTdDhfSWNmLWhickdmdkpOZGF5RVVjekE&usp=sharing) and the task will be on designing and implementing feature extractor for those and - if time allows - experiment with machine learning algorithms to test the effectiveness of such features on the translation of documents. We will use a corpus of **subtitles of TV series** for that.

Word-level quality estimation (Christian Buck)
This project will implement some of the word-level features proposed by the WMT13 QE Shared Task participants, preferably within the QuEst framework. At the end of the week we aim to be state-of-the-art. Or better. [To be joined with general Quest based efforts]

? Domain-Savvy Training for Moses by Default

(Ondřej suggested this but Jon is not coming this time)
http://www.cs.cmu.edu/~jhclark/pubs/amta2012.pdf

Global Lexicon Model training with YASMET

(Francis Tyers -- caveat: He doesn't really know how it works)

- Current GLM training requires MegaM, which is not free/open-source
- YASMET is free/open-source
- Hack the GLM training scripts to use YASMET
- … and then think of efficiency improvements
    - "The training code is a very experimental state. It is very inefficient."
    - (Ondrej and Ales have some code that nicely parallelizes GLM in eman)
- [comment from Chris Dyer - you can implement this much more efficiently using stochastic gradient descent which is just about 5 lines of code; there are also some new variants called AdaGrad that work well with features differ in frequency of occurrency]

Do something with Word Lattices

(Francis Tyers, Bushra Jawaid, Amir Kamran, ...)

- Ondřej would like to link multiple Moses runs with lattices: the first Moses emits search graph that we just need to convert to the input lattice format so that the second Moses can pick it up. The issue are scores, there are probably multiple reasonable things to do with them.
    - A related idea is to implement this moses-sequence (passing lattices from one Moses to the next one) in the moses-server base, so that a single moses process (moses-server as you know it, with multiple configurations loaded) will translate the input step by step. Note that the standard recaser could well be the last in the sequence.

- Fran would like to come up with some (un-,semi-) supervised way of learning the weights when taking into account either lemma/surface form or morpheme -> morpheme MT. - Could work with Armenian, Greenlandic or something *exciting*.

Static Language Model Interpolation and/or Pruning

(Kenneth Heafield)

Many people interpolate language models. IRSTLM does this by loading all the models into RAM in the decoder, which is costly in terms of CPU and memory. SRILM can write a single interpolated model to disk, but it too loads everything in RAM. And the SRILM command line options make grad students cry. Let's write something that can interpolate larger language models.

Common Crawl

(Kenneth Heafield)

What would you do with 6.6 TB of gzipped text and a couple of supercomputers? I downloaded CommonCrawl's text files. These have markup stripped, come in all sorts of languages, and are annotated with the URL they came from. Some usage ideas are big language models in several languages, releasing n-gram counts (the Google n-grams thresholded at count 40), and finding parallel data based on content.

Machine Translation Virtual Kitchen

(Chris Dyer)

The complexity of MT pipelines means that getting started with MT can be difficult for new students, companies, or anyone else who might be interested. This project will create a virtual machine image on Amazon EC2 (which can be exported to be runnable on many different VM hypervisors) that is fully equipped with the latest and greatest in MT "appliances" (Moses, Joshua, cdec, Apertium) and stocked with some of the more popular "ingredients" (WMT parallel corpora, monolingual corpora, parsers, taggers, tokenizers). Building the kitchen and finding the best ingredients will be the main part of the project, but we will also want to write a good recipe book to help folks get started with MT.

This project is inspired by:
http://www.cs.cmu.edu/~fmetze/interACT/Publications_files/publications/cri-showandtell.pdf

Common Phrase Table and API

(Kenneth Heafield, Marcin Junczys-Dowmunt)

Why does each decoder need a different phrase table implementation? If somebody implements a good phrase table, everybody should use it. If somebody wants to implement a decoder, they shouldn't need to write and maintain their own phrase table. Let's settle on a common API

across decoders and make one phrase table to rule them all (including phrase-based and parsing-based). Probably boils down to a more or less clever string container. Have to think about that some more. Maybe only one very restricted interface (sentence in - phrases/rules out)? Should correspond performance-wise with the Minimalistic Decoder.

Improving source code quality

(Jacob Dlougach)

. This project involves some “low-level” engineering

- Some parts of moses source code can benefit from refactoring (to make them more comprehensible and more “logical”).
- Sometimes Moses crashes without giving any clues to what has happened. Thus, it would also be useful to add some runtime checks to diagnose problems early before they lead to unexpected crashes in training or decoding.
- Boostification where it can help
- What’s the current take on C++11 features? [CB]
- Boost ptr_vector etc instead of RemoveAllInColl -Kenneth
- Unit testing - and refactoring where necessary to make the code unit testable - Barry

Extending a discriminative monolingual aligner to support MT word alignment

(Xuchen Yao, Johns Hopkins University)

Below is a screenshot of the state-of-the-art word aligner for English sentence pairs.

The jacana aligner is discriminatively trained with a lot of language-aware features. It is highly modularized and **we need your help to write some feature functions to support other language pairs** (German<->English, Czech<->English, German <->Czech, etc). This can be very easily done as long as:

- there are labeled word alignment training data
- you know both the source and the target language

If the collaboration is successful, we will altogether submit a demo paper for ACL 2014! If you are interested in this project, feel free to add your name under this project title or shoot me an email!
Following up, other language-independent side coding practice that's helpful to this project includes:

- incorporating jwktl, the java interface to wiktionary, which provides multilingual translation/mapping of different languages
- incorporating some version of "EuroWordNet"

Morphology for Moses

(Kristina Toutanova, Alex Fraser, Aleš Tamchyna, Ondřej Bojar, Fabienne Braune)
In short, the goal is to reimplement the paper A Discriminative Lexicon Model for Complex Morphology (Minwoo Jeong, Kristina Toutanova, Hisami Suzuki, Chris Quirk; 2010. Proc. of

AMTA) using Vowpal Wabbit.

Goal: implement a discriminative lexicon which scores the translations of words into morphologically rich languages (Jeong, Toutanova, Suzuki, Quirk AMTA 2010) using Vowpal Wabbit (Langford).

What we have so far: we have tightly integrated the Vowpal Wabbit (VW) classifier library into Moses so that it can be used during decoding. VW is a classifier aimed at massive data streaming scenarios, it is very fast and uses the hashing trick to minimize memory usage. Up until now we have implemented a discriminative phrase lexicon. Training is conducted using text files and supports multithreading. Training examples are created for each extracted phrase-pair token from the entire training data (not the dev set). We predict the English phrase given the French phrase and the rest of the French sentence, which is sometimes referred to as Phrase Sense Disambiguation (PSD, Carpuat and Wu 2007). Decoding is currently integrated as a feature function outputing log probabilities (similar in spirit to the feature function p(e|f) implemented in the Moses phrase table). This feature function is integrated into pruning and future cost. We have also implemented it into hierarchical decoding (Braune et al, forthcoming), but that will not be addressed at the MTM. For more details on phrase-based PSD using VW see (Tamchyna et al 2014, Carpuat et al 2012).

What will be done on the project in Prague: we will add support to our Vowpal Wabbit interface for scoring at the word level, rather than at the phrase level. This will involve changing training to extract examples which are words or minimal phrases (we will experiment with both of these options). It will also involve changing decoding to call the VW classifier multiple times for each phrase-based hypothesis extension, once for each target word or minimal translation unit in the phrase-pair. We plan to experiment on English to Czech translation and take advantage of the availability of very high quality analysis of English and Czech. There will also be significant work on defining features which capture inflectional decisions for the English to Czech language pair.

Forest Rescoring for MIRA

(Matt Post)

Cherry & Foster contributed a batch MIRA implementation to Moses, which is also used by Joshua. However, the k-best lists output by these decoders (also cdec?) are model-best lists, which represent only a very small fragment of the search space. This is in contrast to David Chiang's MIRA papers, which provide a richer discriminative environment to MIRA by also extracting "hope" and "fear" k-best lists from the forests. These are defined as:

- hope: model score + BLEU score
- fear: model score - BLEU score

These lists can be extracted by rescoring the hypergraph with a BLEU approximation that factors over the edges of the hypergraph. In this project, we will do just that, in hopes of improving tuning with MIRA.

I will be heading up forest rescoring in Joshua. Time-permitting, I'll move to Moses. It would be helpful to have someone familiar with Moses to work on that piece. I would also happily accept help on the Joshua side.

References: (See especially Chiang et al. (2009), Cherry & Foster (2012), and Chiang (2012))

- Crammer, Koby and Dekel, Ofer and Keshet, Joseph and Shalev-Shwartz, Shai and Singer, Yoram. Online passive-aggressive algorithms. JMLR, 2006.
- Watanabe, Taro and Suzuki, Jun and Tsukada, Hajime and Isozaki, Hideki. Online large-margin training for statistical machine translation. EMNLP, 2007.
- Chiang, David and Marton, Yuval and Resnik, Philip. Online Large-Margin Training of Syntactic and Structural Translation Features. EMNLP, 2008.
- Chiang, David and Knight, Kevin and Wang, Wei. 11,001 new features for statistical machine translation. NAACL, 2009.
- Cherry, Colin and Foster, George. Batch Tuning Strategies for Statistical Machine Translation. NAACL, 2012.
- Chiang, David. Hope and fear for discriminative training of statistical translation models. JMLR, 2012.

Social Media MT toolkit

(Wang Ling, Chris Dyer)

Goal: Adaptation of MT components for translation on Social Media. Our goal is to adapt MT components in order to improve their performance in Social Media data.

Why: The non-standard language used in Social Media (abbreviations, jargon and orthographic error etc...) generally cause additional challenges in MT. Recently, we published an

automatically extracted microblog parallel corpora at available http://www.cs.cmu.edu/~lingwang/microtopia/, which can be used to obtain improvements on such data. However, many existing components in the MT pipeline do not work well on such data. Examples:

- Tokenizer/Detokenizer - Most tokenizers for MT are written for other domains without considering the kind of phenomena that occur in Social Media. For instance, the tokenizer used in Moses does not take into account the following:

Example	Original	Moses tokenizer output
Http references	New blog post: Wiz Khalifa Announces “High School 2″ With **Snoop Dogg** http://bit.ly/16LPEpp	new blog post : wiz khalifa announces “ high school 2 ″ with snoop dogg **http : / / bit.ly / 16lpepp**
@ references	Listen to **@TerraceMartin**'s '3ChordFold' ft.	listen to **@ terracemartin** 's ' 3chordfold ' ft .
# tags	With Kurupt @ **Snoop Dogg** live **#DOGGPOUND** http://instagram.com/p/c7ikHsCk8A/	with kurupt @ snoop dogg live **# doggpound** http : / / instagram.com / p / c7ikhsck8a /
emoticons	I miss you seth .." miss you to buddy **:)**	i miss you seth .. " miss you to buddy **:)**
….	…..	…..

- MT models - Most models assume that words with different forms are different words entirely. This is not true in Social Media, since there are many different variations of the same word. For instance, the words “gonna”, “gunna”, “gonnnna” and “going to” have similar if not the same meaning, but the alignment and translation models would treat them differently.
- Evaluation - As with the models, if the proposed translation uses one word form, such as “gonna” and the reference has another, such as “going to”, the metric will fail to acknowledge even a partial score for the translation.

What we will bring: The microblog corpora will given to participants, and we will provide scripts to build and test the baseline using moses. From there participants can address the issues above or come up with their own ideas.

Inline Tag Handling

(Achim Ruopp)

HTML and other formats allow for structured inline tagging for formatting and placeholders.

When translating tagged sentences with Moses, the tags need to be protected/passed through the decoder, or removed before decoding and projected onto the target sentence after decoding. The M4Loc project integration component parses different formats using the Okapi toolkit and allows to protect tags during decoding and project tags from source to target using phrase alignment info emitted by the decoder.

The quality of the tag handling can be improved by using word alignment info now available from the decoder, using a union of tag coverage over the output tokens or other methods. The goal for MTM would be to implement one or two new approaches for tag handling and compare the quality of the tag handling to existing approaches. Potentially we can also come up with a better method to judge the quality of tag handling than using standard MT metrics over a mix of tokens and tags.

Earlier work:

Tomá Hudík and Achim Ruopp, The Integration of Moses into Localization Industry, Proceedings of the 15th International Conference of the European Association for Machine Translation, May 2011, Leuven, Belgium

Eric Joanis, Darlene Stewart, Samuel Larkin and Roland Kuhn, Transferring Source Tags to the Target Text in Statistical Machine Translation: A Two-Stream Approach, September 2013, MT Summit XIV Workshop on Post-editing Technology and Practice, Nice, France

Du, Jinhua, Johann Roturier and Andy Way, TMX markup: a challenge when adapting SMT to the localisation environment. In: EAMT 2010 14th Annual Conference of the European Association for Machine Translation, Saint-Raphaël, France.

Arda Tezcan and Vincent Vandeghinste, SMT-CAT integration in a Technical Domain: Handling XML Markup Using Pre & Post-processing Methods, Proceedings of the 15th International Conference of the European Association for Machine Translation, May 2011, Leuven, Belgium

Lattice rescoring into Moses

(Loïc Barrault, Fethi Bougares, Francis Tyers)

Goal : Moses online Lattice rescoring

1.1. Generate the search space (fast decoding pass)
1.2. Update the generated search space using more complicated models
 1.2.1. neural network LM -> CSLM already integrated, needs cleaning and push into moses
 1.2.2. higher order n-gram LM
 1.2.3. additional feature functions … etc.
1.3. Determine the correct tuning process for such framework

Description : This project will implement an integrated lattice/nbest rescoring process into moses. The rescoring process can be done by modifying already used models (from decoding pass) or by adding new models (more sophisticated or adapted ones). The latter can be seen as multipass SMT.

http://github.com/kpu/lazy does lattice rescoring as a special case of hypergraph rescoring. It's split into a library and a standalone wrapper. The library half is already in Moses under search. I also have modifications to output a pruned rescored lattice in lieu of a K-best list. -KennethThis project probably links with "Do something with Word Lattices" from Fran Tyers.

Bundling the TrTok Trainable Tokenizer with Moses

(Jiří Maršík)

I will be taking the TrTok tokenizer (PDF, GitHub, used in CzEng) and trying to ease its distribution with Moses. The tokenizer has hefty compile-time and runtime dependencies, some of them not even available in the biggest package repositories. I will be looking at ways for automating the deployment proces1s so others can benefit from (slightly) more accurate tokenization in their MT pipelines.

Help from those knowledgeable in the Moses build pipeline will be welcome.

Moses scripts for automatically generating IDE project files

(Lane Schwartz, working remotely)

The XCode project for Moses is way out of date, and it is hard to keep it up to date. Lane Schwartz has been working to bring the project file up to date. This proposal would remove the project file from git revision control and instead add some script or other build process to automatically generate the project file. This would solve the problem of having to make manual corrections to the project file whenever someone adds, removes, or renames a file or dependency. Arguably the best place for this sort of IDE project file generation functionality would be in bjam, since it already knows about all of the source files, dependencies, and executable targets.

XCode would be the first IDE to be targeted, but it would be nice to have this sort of thing available for other popular IDEs. Eclipse and CodeBlocks project files might be worth looking at, since they both support Linux, Mac OS X, and Windows.

MTSpell - Spelling correction for post-edited MT

(Christian Buck, Francis Tyers)
Idea: Using source side information for spell checking of post-edited MT output.

MT makes many terrible errors but mistyping is rarely among these. That changes once a post-editor mangles with the text to transform it into a proper translation. A very simple spelling correction would find words that don't occur in a dictionary and suggest similar words based on Edit distance and soundex similarity. A more complex spelling correction might apply a language model - possibly the one used to generate the MT suggestion - to score that lattice of correction possibilities and suggest the high scoring alternatives. But we can also use cues from the source side, e.g. suggesting only corrections that also occurred in the MT search graph.

The goal of this project is to rapidly develop a spelling correction that used information from various sources including character n-gram models, word n-gram models, external spell checkers, phonetic similarity, edit distance under some model of typical typing errors and the search graph that was used to generate the original suggestion.

This project already has a github repo containing a prototype that can produce a lattice of possible corrections which be scored with Ken's standalone decoder.

-Fran would like this to support voikko, which implements finite-state spell-checkers for morphologically complex languages (among other things)

Internal tree structure for GHKM rules in Moses

(Phil Williams, Maria Nadejde)
The GHKM algorithm produces synchronous grammar rules that map source strings to target tree fragments (with variables). Moses' GHKM rule extractor reduces these to SCFG rules by discarding the internal tree structure of the tree fragment. Whilst the internal structure isn't required for decoding, it may be useful for defining feature functions or for generating "complete" parse trees as output.

This project will modify Moses' string-to-tree pipeline so that the (sometimes ambiguous) internal tree structure is recorded for each extracted SCFG rule and made available within the decoder.

+1 from Alex Fraser, I'd love to have this. (Eventually it would be nice to have tree fragment matching in the source too, then you have STSG)

https://docs.google.com/document/d/1n6CYSd_FnN10lvDaOvANsY14-cOxYOlUVKT41MtWiJw/edit

Sparse Features for Reordering

(Barry Haddow)

This is based on the paper “Improved Reordering for Phrase-Based Translation using Sparse Features” by Colin Cherry (NAACL 2013).

Sparse features in MT have been, to an extent, a solution in search of a problem. One such possible problem is reordering, and the Cherry paper shows that sparse features can be useful, and better than training a discriminative reordering model and using its output as a feature. The idea is that reordering models can be trained to maximise an objective we care about (like bleu) instead of one we don’t care about (log-likelihood of reordering model).

Improved Reordering for Phrase-Based Translation using Sparse Features

The aims of this project are:

- Reimplement the Cherry features in Moses - at the moment they are only available in the super-secret NRC decoder
- See if they work on a smallish model in a different language pair (de-en, probably)
- Look for possible extensions/variations - e.g. incorporating dependency parse
- Extend the Moses implementation of batch mira to work with lattices - said to be useful for training these features - see Matt’s project
- Look at train/evaluating with a reordering-sensitive measure like LRScore
- See if these or similar features can be applied to hiero models
- Exercise the new Moses feature function interface!

Reference implementation of word-based decoder

(Lane Schwartz, working remotely)

We celebrate the 20th anniversary of the IBM word-based translation models with a grand total of zero open source word-based machine translation systems. The seminal IBM papers claimed that a forthcoming paper would describe their decoder, but no such paper was ever published.

It would be a useful point of comparison to directly compare the original statistical MT models against the current in the state-of-the-art.

This project will use algorithms described in the relevant IBM patent, which has now expired, to implement an open source word-based statistical machine translation decoder.

- Implement a decoder that uses IBM Model 1 as the TM. KenLM will be used as the LM.
- As time permits, extend this decoder to use other word-based TMs.

To do:

1. Email Lane Schwartz (dowobeha@gmail.com) to let him know you're on the project
2. Take a look at U.S. Patent 5,477,451 (https://www.google.com/patents/US5477451)
3. Lane has started writing up the word-based algorithm. We need to finish this. The current blocking point is determining how to organize and initialize the priority queues. Section 14.3 of the patent states "In theory, there is a single priority queue for each subset of positions in the source structure. … In practice these priority queues are initialized only on demand, and many less than the full number of queues possible are used in the hypothesis search."

Can coreference help in translating other words than pronouns?

(Michal Novák)

How to translate the verb **cut** into Czech in the following sentence?
"He bought *a bread* and **cut** *it* into pieces."
Replacing the pronoun (*it*) with the correct antecedent (*a bread*) gives us more information to resolve the correct translation variant of the verb **cut**.Possible translations of **cut**:

- **cut** prices -> **snížit** ceny
- **cut** bread -> **krájet** chleba
- **cut** forest -> **kácet** les
- **cut** paper -> **rezat/stříhat** papír

We plan to integrate the Sieve coreference resolver from the Stanford CoreNLP system into the TectoMT system, re-train the translation models for the English-Czech language pair with the coreferential features and see what happens :)
Perl programming.