MT Marathon 2016 (Prague)

Proposed Projects

Start by **sharing the document with yourself** (so that Google records edits well).
Then simply 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.

Please prepare slides for your project **midweek report**.
If possible, commit the PDF of the slide to the Marathon SVN:

```
svn co https://svn.ms.mff.cuni.cz/svn/mtmarathon-2016/trunk/projects
# into the directory midweek-reports
```

If you don’t have an account at our svn server yet, your username is most likely name.surname and your password is most likely the (lowercase) string between ‘@’ and the first dot in your e-mail address, e.g. ‘gmail’.
Send an e-mail to Ondřej Bojar (bojar@ufal.mff.cuni.cz) with the output of the command

```
htpasswd -snb your.name.surname your-new-password
```

for a password reset.

List of Projects (automatically generated)

Sample Project

(John the Proposer; include e-mail if you want to)

- The bright idea.
- The goal.
- Prospective participants: John the Proposer

NMT Experiments with Neural Monkey

(Jindra Helcl, Ondrej Bojar; {helcl,bojar}@ufal.mff.cuni.cz)

- Neural Monkey is a modular multi-source NMT tool implemented in TensorFlow.
- The goal of the project (breaking up into several sub-projects) is to extend Neural Monkey in several ways:

    - “Factored input”, i.e. c (in each input stream)
    - Multi-task learning, like http://arxiv.org/abs/1511.06114 i.e. adding support for multiple decoders, each targeting a different task, learning from a stream of examples presented as tab-delimited file.
        - First some number of input columns. (Can these be empty??)
        - Then some number of output columns. Any subset of these can be empty, causing a ‘dropout-like’ behaviour, i.e. not updating any of the cells in that part of the network.
    - https://arxiv.org/abs/1601.01073 Multi-Way, Multilingual NMT with shared attention
    - Other-than sequence decoder, e.g. predicting a parse tree.
        - Neural Monkey can so far produce only one simple sequence of tokens.
        - It would be interesting to use the architecture for predicting other structures as well; in particular with multi-task learning (the previous sub-project), we could learn to simultaneously parse and translate etc. A possible solution would be to predict the sequence of transitions.
- Prospective participants: *enter yourself, possibly mentioning the subproject you’re most interested in*
    - Jindra Helcl (all, with focus on something)
    - Martin Popel (all subtasks, but mostly factored input)
    - Eleftherios Avramidis (quality estimation/indications)
    - Jernej Vicic (factored input)

Domain specific translations: Moses vs. NMT

(Roman Sudarikov, only remote; Tom Kocmi; Ondrej Bojar?)
See https://groups.google.com/forum/#!forum/domain-specific-translations--moses-vs-nmt

- Moses is well-known PBSMT
- NMT are known to beat Moses-based systems on WMT2016 for general domain translations
- The goal of the project is to compare NMT (Nematus or Neural Monkey) in a domain-specific task:
    - Data will be IT domain from QT21 project
    - Participants can compare both proposed NMT systems or select one or suggest their own
- During the project you can learn:
    - How to setup NMT system
    - How to setup and manage Moses using Eman experiments management tool
- Prospective participants: *enter yourself, possibly mentioning the NMT you’re most interested in*
    - Roman Sudarikov (Nematus, Neural Monkey)

Compressed Suffix Trees for Language Modelling

(Matthias Petri, Trevor Cohn)

Please email Matthias Matthias.Petri@unimelb.edu.au to meet and get started.

Compressed Suffix Trees (CSTs) are a recent compact representation of Suffix Trees which allow indexing large amounts of data with limited resources. CSTLM is a language model available at https://github.com/mpetri/CSTLM/ built on top of CSTs which utilises the search capabilities of the CST to provide infinite-context language models with modest memory requirements for both character and word-level language models. The space usage of CSTLM is comparable (often much smaller) than other state-of-the-art models.

The current implementation of CSTLM is still of "research" quality but returns Modified Kneser-Ney perplexities matching those return by popular LM toolkits such as KenLM.

In this project we propose the following tasks:

- Optimise the preliminary implementation of CSTLM into moses available at https://github.com/mpetri/mosesdecoder to fully utilise the query functionality of the LM. This includes development along the lines of optimising the interface between moses and the LM, and algorithmic issues of how best to decode in terms of pruning heuristics, future cost etc. Finally, development of algorithms for LM integration with hierarchical phrase-based decoding.
- Development and experimentation with unbounded language models over words and other factors, both at the word and byte (~character) level, with a special focus on morphologically rich languages.
- Evaluate CSTLM in a "fair" experimental setting against state-of-the-art LMs such as KenLM in terms of (1) construction costs (2) raw query performance (3) query performance in the context of moses
- Enhance CSTLM by including recent semi-external suffix array construction algorithms to reduce the space usage at construction time

References: Preliminary evaluations and details regarding the implementation of a LM using CSTs have been carried out in the following publications:

- Ehsan Shareghi, Matthias Petri, Gholamreza Haffari, Trevor Cohn:

Compact, Efficient and Unlimited Capacity: Language Modeling with Compressed Suffix Trees. EMNLP 2015: 2409-2418: http://aclweb.org/anthology/D/D15/D15-1288.pdf

- Ehsan Shareghi, Matthias Petri, Gholamrezaffari, Trevor Cohn:

Fast, Small and Exact: Infinite-order Language Modelling with Compressed Suffix Trees
TACL 2016: to appear (available on arXiv: http://arxiv.org/abs/1608.04465)

Requirements: C++

Prospective participants: Matthias Petri,

Non-perplexity neural MT objectives

(Kenneth Heafield, Rico Sennrich)
Perplexity is a bad training objective. Training sees gold contexts but decoding makes mistakes, so there's a mismatch between training and usage. On a related note, normalization means that a "lost" network has no way to signal that it is lost, creating a label bias problem. Fortunately, there has been a variety of work on solving that:

- Sequence Level Training with Recurrent Neural Networks
- Minimum Risk Training for Neural Machine Translation
- Globally Normalized Transition-Based Neural Networks (in parsing)
- Sequence-to-Sequence Learning as Beam-Search Optimization
- Reward Augmented Maximum Likelihood for Neural Structured Prediction

In this project, we'll pick one (or more depending on interest) objective and implement it in an otherwise-state-of-the-art neural MT system. This could be done in Nematus (https://github.com/rsennrich/nematus/) or your favorite neural toolkit, possibly based on Monkeys.
Requirements: Python, Most will be taught in the Marathon
Desired: GPU and toolkit experience
Prospective participants: Marcin will help (debatable), Julian Hitschler