JoshuaProposal

Joshua Proposal

Abstract

Joshua is an open-source statistical machine translation toolkit. It includes a Java-based decoder for translating with phrase-based, hierarchical, and syntax-based translation models, a Hadoop-based grammar extractor (Thrax), and an extensive set of tools and scripts for training and evaluating new models from parallel text.

Proposal

Joshua is a state of the art statistical machine translation system that provides a number of features:

- Support for the two main paradigms in statistical machine translation: phrase-based and hierarchical / syntactic.
- A sparse feature API that makes it easy to add new feature templates supporting millions of features
- Native implementations of many tuners (MERT, MIRA, PRO, and AdaGrad)
- Support for lattice decoding, allowing upstream NLP tools to expose their hypothesis space to the MT system
- An efficient representation for models, allowing for quick loading of multi-gigabyte model files
- Fast decoding speed (on par with Moses and mtplz)
- Language packs — precompiled models that allow the decoder to be run as a black box
- Thrax, a Hadoop-based tool for learning translation models from parallel text
- A suite of tools for constructing new models for any language pair for which sufficient training data exists

Background and Rationale

A number of factors make this a good time for an Apache project focused on machine translation (MT): the quality of MT output (for many language pairs); the average computing resources available on computers, relative to the needs of MT systems; and the availability of a number of high-quality toolkits, together with a large base of researchers working on them.

Over the past decade, machine translation (MT; the automatic translation of one human language to another) has become a reality. The research into statistical approaches to translation that began in the early nineties, together with the availability of large amounts of training data, and better computing infrastructure, have all come together to produce translations results that are “good enough” for a large set of language pairs and use cases. Free services like Bing Translator and Google Translate have made these services available to the average person through direct interfaces and through tools like browser plugins, and sites across the world with higher translation needs use them to translate their pages through automatically.

MT does not require the infrastructure of large corporations in order to produce feasible output. Machine translation can be resource-intensive, but need not be prohibitively so. Disk and memory usage are mostly a matter of model size, which for most language pairs is a few gigabytes at most, at which size models can provide coverage on the order of tens or even hundreds of thousands of words in the input and output languages. The computational complexity of the algorithms used to search for translations of new sentences are typically linear in the number of words in the input sentence, making it possible to run a translation engine on a personal computer.

The research community has produced many different open source translation projects for a range of programming languages and under a variety of licenses. These projects include the core “decoder”, which takes a model and uses it to translate new sentences between the language pair the model was defined for. They also typically include a large set of tools that enable new models to be built from large sets of example translations (“parallel data”) and monolingual texts. These toolkits are usually built to support the agendas of the (largely) academic researchers that build them: the repeated cycle of building new models, tuning model parameters against development data, and evaluating them against held-out test data, using standard metrics for testing the quality of MT output.

Together, these three factors—the quality of machine translation output, the feasibility of translating on standard computers, and the availability of tools to build models—make it reasonable for the end users to use MT as a black-box service, and to run it on their personal machine.

These factors make it a good time for an organization with the status of the Apache Foundation to host a machine translation project.

Current Status

Joshua was originally ported from David Chiang’s Python implementation of Hiero by Zhifei Li, while he was a Ph.D. student at Johns Hopkins University. The current version is maintained by Matt Post at Johns Hopkins’ Human Language Technology Center of Excellence. Joshua has made many releases with a list of over 20 source code tags. The last release of Joshua was 6.0.5 on November 5th, 2015.

Meritocracy

The current developers are familiar with meritocratic open source development at Apache. Apache was chosen specifically because we want to encourage this style of development for the project.

Community

Joshua is used widely across the world. Perhaps its biggest (known) research / industrial user is the Amazon research group in Berlin. Another user is the US Army Research Lab. No formal census has been undertaken, but posts to the Joshua technical support mailing list, along with the occasional contributions, suggest small research and academic communities spread across the world, many of them in India.

During incubation, we will explicitly seek to increase our usage across the board, including academic research, industry, and other end users interested in statistical machine translation.

Core Developers

The current set of core developers is fairly small, having fallen with the graduation from Johns Hopkins of some core student participants. However, Joshua is used fairly widely, as mentioned above, and there remains a commitment from the principal researcher at Johns Hopkins to continue to use and develop it. Joshua has seen a number of new community members become interested recently due to a potential for its projected use in a number of ongoing DARPA projects such as XDATA and Memex.

Alignment

Joshua is currently Copyright (c) 2015, Johns Hopkins University All rights reserved and licensed under BSD 2-clause license. It would of course be the intention to relicense this code under AL2.0 which would permit expanded and increased use of the software within Apache projects. There is currently an ongoing effort within the Apache Tika community to utilize Joshua within Tika's Translate API, see TIKA-1343.

Known Risks

Orphaned products

At the moment, regular contributions are made by a single contributor, the lead maintainer. He (Matt Post) plans to continue development for the next few years, but it is still a single point of failure, since the graduate students who worked on the project have moved on to jobs, mostly in industry. However, our goal is to help that process by growing the community in Apache, and at least in growing the community with users and participants from NASA JPL.

Inexperience with Open Source

The team both at Johns Hopkins and NASA JPL have experience with many OSS software projects at Apache and elsewhere. We understand "how it works" here at the foundation.

Relationships with Other Apache Products

Joshua includes dependences on Hadoop, and also is included as a plugin in Apache Tika. We are also interested in coordinating with other projects including Spark, and other projects needing MT services for language translation.

Developers

Joshua only has one regular developer who is employed by Johns Hopkins University. NASA JPL (Mattmann and McGibbney) have been contributing lately including a Brew formula and other contributions to the project through the DARPA XDATA and Memex programs.

Documentation

Documentation and publications related to Joshua can be found at joshua-decoder.org. The source for the Joshua documentation is currently hosted on Github at https://github.com/joshua-decoder/joshua-decoder.github.com

Initial Source

Current source resides at Github: github.com/joshua-decoder/joshua (the main decoder and toolkit) and github.com/joshua-decoder/thrax (the grammar extraction tool).

External Dependencies

Joshua has a number of external dependencies. Only BerkeleyLM (Apache 2.0) and KenLM (LGPL 2.1) are run-time decoder dependencies (one of which is needed for translating sentences with pre-built models). The rest are dependencies for the build system and pipeline, used for constructing and training new models from parallel text.

Apache projects:

- Ant
- Hadoop
- Commons
- Maven
- Ivy

There are also a number of other open-source projects with various licenses that the project depends on both dynamically (runtime), and statically.

GNU GPL 2

- Berkeley Aligner: https://code.google.com/p/berkeleyaligner/

LGPL 2.1

- KenLM: github.com/kpu/kenlm

Apache 2.0

- BerkeleyLM: https://code.google.com/p/berkeleylm/

GNU GPL

- GIZA++: http://www.statmt.org/moses/giza/GIZA++.html

Required Resources

- Mailing Lists
    - private@joshua.incubator.apache.org
    - dev@joshua.incubator.apache.org
    - commits@joshua.incubator.apache.org
- Git Repos
    - https://git-wip-us.apache.org/repos/asf/joshua.git
- Issue Tracking
    - JIRA Joshua (JOSHUA)
- Continuous Integration
    - Jenkins builds on https://builds.apache.org/
- Web
    - http://joshua.incubator.apache.org/
    - wiki at http://cwiki.apache.org

Initial Committers

The following is a list of the planned initial Apache committers (the active subset of the committers for the current repository on Github).

- Matt Post (post@cs.jhu.edu)
- Lewis John McGibbney (lewismc@apache.org)
- Chris Mattmann (mattmann@apache.org)
- Henry Saputra (hsaputra@apache.org)
- Tommaso Teofili (tommaso@apache.org)
- Tom Barber (magicaltrout@apache.org)

Affiliations

- Johns Hopkins University
    - Matt Post
- NASA JPL
    - Chris Mattmann
    - Lewis John McGibbney

Sponsors

Champion

- Chris Mattmann (NASA/JPL)

Nominated Mentors

- Paul Ramirez
- Lewis John McGibbney
- Chris Mattmann
- Tom Barber
- Henri Yandell

Sponsoring Entity

The Apache Incubator