LLTools:
Machine Learning for Human Language Processing

Human Language Technology Group
MIT Lincoln Laboratory
Lexington, MA 01740
dagli@ll.mit.edu

Abstract

Machine learning methods in Human Language Technology have reached a stage of maturity where widespread use is both possible and desirable. The MIT Lincoln Laboratory LLTools software suite provides a step towards this goal by providing a set of easily accessible frameworks for incorporating speech, text, and entity resolution components into larger applications. For the speech processing component, the pySLGR (Speaker, Language, Gender Recognition) tool provides signal processing, standard feature analysis, speech utterance embedding, and machine learning modeling methods in Python. The text processing component in LLTools extracts semantically meaningful insights from unstructured data via entity extraction, topic modeling, and document classification. The entity resolution component in LLTools provides approximate string matching, author recognition and graph-based methods for identifying and linking different instances of the same real-world entity. We show through two applications that LLTools can be used to rapidly create and train research prototypes for human language processing.

1 Introduction

The commoditization of big-data analytics and the emergence of open-source and commercial Machine Learning as a Service (MLaaS) platforms put a high premium on developing algorithmic components that are fast, easy-to-use and accessible. Such components are invaluable as they enable rapid prototyping of large-scale systems through easy integration into existing machine learning pipelines. Algorithms and tools which are not able to fit into this paradigm are at a distinct disadvantage when it comes to adoption and further development in this new machine learning environment.

Human language technology is an interdisciplinary field that comprises signal processing, computational methods and machine learning methods for analyzing, producing and/or modifying speech and text. Despite being a mature field of study, there remains a surprising lack of general purpose, end-to-end tool sets which can support the needs of human language processing in the emerging machine learning paradigms. And existing popular packages such as NLTK [1], gensim [2] and SyntaxNet [3] often focus on specific data domains and application scenarios.

In this paper, we introduce a tool set, LLTools suite (http://github.com/mitll). We seek to provide to the machine learning community a more general-purpose set of tools which can support

---

This work was sponsored by the Defense Advanced Research Projects Agency under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States Government.

processing multiple forms of human language data (e.g., speech and text) and downstream applications (e.g., entity resolution and search). The rest of the paper is organized as follows. We begin by outlining our architectural vision in Section 2. Sections 3-5 provide detailed profiling of specific component in the LLTools suite, including speech processing, natural language processing and entity resolution methods. Lastly, we present a few real-world examples of LLTools in action in Section 6.

2 Architecture for Human Language Tools

The LLTools suite provides the capability to build practical, real-world systems for processing, structuring and performing inference on raw human language data (speech and text). It is designed around efficient implementations and interfaces to provide easy-accessibility for rapid prototyping of full human language processing systems. An overview of the LLTools architecture can be seen in Figure 1 below.

![LLTools System Architecture](image)

We classify component tools into two separate yet related categories: unstructured and structured data analysis. Unstructured data analysis refer to those tools that take as input raw speech or text data and produce as output per-document enrichments. Two examples of such enrichments are entity extraction from text and speaker detection from speech. Outputs from these tools can either be consumed directly on their own or be input for downstream structured data analysis tools.

Structured data analysis refer to those tools that take as input enriched raw data (e.g., entities, topics or speakers) and perform cross-document (corpus-level) analysis. An example of such an analysis is entity resolution and search. As these tools look to ask questions across documents within a corpus (or even across corpora), efficient matching and search tools are included as well.

In the following sections, we describe the LLTools suite with respect to each component of the proposed Human Language Technology architecture. The components are organized by raw signal processing, unstructured text processing, and entity resolution from processed and structured data. In addition, we provide some example applications of using LLTools on real-world human language technology data and problems.

3 Speaker, Language and Gender Recognition (SLGR)

A common task in speech processing is to take raw signal data and convert it to structured form. The pySLGR (Speaker, Language, Gender Recognition) tool implements standard methods for extracting this meta-data.

The pySLGR system is organized in two layers, see Figure 2. In the first layer, the basic functionality for SLGR is implemented as a C++ toolkit. In the second Cython layer, an interface to the C++ tools is provided via a Python API. The API then exposes a set of SLGR capabilities, for example, signal processing (i.e., LLSignal), feature extraction and normalization (i.e., LLFeatures), and model building and scoring (i.e., iVector, GMMModel, SuperVector).
The front end processing of raw signal data in pySLGR consists of LLSignal and LLFeatures. LLSignal provides basic signal processing techniques including FIR filtering, resampling, and load/save functionality. LLFeatures provides standard mel-cepstral or filter bank features. Compensation such as feature norming, feature warping, and RASTA are also implemented [4].

For classification, pySLGR implements several standard models. For vector representations, the system implements GMM (Gaussian mixture model) supervector extraction [5] and the reduced dimension i-vector [6] method. Additionally, pySLGR contains GMM training and scoring. Pre-built models are also available for gender, language, and speaker recognition.

4 Natural Language Processing

To derive semantically meaningful insights from such highly-unstructured human language data, practical systems must have available a suite of pre-processing tools to begin structuring the data for use in downstream components. These tools can be used to provide low-level (e.g., entity), mid-level (e.g., topic) and high-level (e.g., classification) document enrichments. In this section, we describe a set of tools that perform such data structuring.

4.1 MITIE

An important first-step in basic data structuring is to extract entities (e.g., people, places, organization, etc.) from the raw text. The MIT Information Extraction (MITIE) project provides state-of-the-art information extraction tools for performing named entity extraction and binary relation detection, as well as tools for training custom extractors and relation detectors.

MITIE makes use of several state-of-the-art techniques such as distributional word embeddings [7] and Structural Support Vector Machines [8]. MITIE also offers several pre-trained models to provide varying levels of support for both English and Spanish. MITIE is written on top of dlib [9] in C++, but bindings for other software languages (i.e., Python, R, Java, C, and MATLAB) are provided for the ease of rapid application development.

4.2 Topic

Unsupervised enrichments from corpus-level statistics provide meaningful mid-level annotations that are useful for a variety of downstream applications. Topic clustering is a powerful example of such an enrichment. It facilitates the tagging, filtering and grouping of documents based on lexicographical similarity. It is also highly valuable in situations where manual annotation of data is unavailable or highly infeasible.

MIT Lincoln Laboratory’s Topic Clustering system provides an open-source, efficient implementation of a topic modeling pipeline. It includes routines for text normalization and soft-classification using the Probabilistic Latent Semantic Analysis (pLSA) algorithm [10]. Results are stored in human/machine readable files that can be used for browsing or integration into further stages.

4.3 LLClass

High-level semantic categories for text (e.g., language being spoken, sentiment of a discussion, etc.) can be used for raw data structuring in the pre-processing stage. Lincoln Laboratory’s text classification tool (LLClass) enables such enrichments by providing both pre-trained and customizable text categorization tools.
LLClass provides capabilities for feature extraction and classification of documents by their lexicographical content. It includes several state-of-the-art tools for implementation of the Margin-Infused Relaxation Algorithm (MIRA) \[1\] that allows for online-training of max-margin classifiers. LLClass has been used for a number of data categorization tasks and is distributed with a state-of-the art multi-class Twitter Language Identification system.

5 Entity Resolution

Human language data can be structured for analysis by extracting a content graph of entities and their relations. This process is usually performed in three stages as shown in Figure 3. First, information is extracted from raw input (speech, text). For text, named mentions of entities (people, places, locations) are found in the data; see Section 4. For speech, a vector representation of the speaker of interest is extracted \[6\]; see Section 3. Second, the extracted information is stored and indexed, and links between entities are formed based on similarity \[12\] or co-occurrence \[13\]. Third, multiple mentions (“Bob Smith”, “Bob S.”) are resolved to the same entity using entity resolution. Resolution is usually cast as a verification (detection problem) or a clustering problem depending on the application. In this section, we discuss tools that implement the entity resolution process.

5.1 LLString

The most straightforward form of entity resolution is based on approximate string matching of profile entries, e.g., an entity’s full name or the username. A common strategy for users is to add a few extra characters to their username on one platform to obtain a username on another platform.

Our configurable pipeline for approximate string matching is as follows. First, LLString provides text normalization to convert unusual UTF-8 characters to a standardized form, eliminate emojis, eliminate emoticons, and remove markup. Second, LLString includes a set of standard soft string matching methods, i.e., Levenshtein, Jaro, Jaro-Winkler, and Jaro-Winkler with soft-TFIDF; see, for example \[14\]. The TFIDF component is trainable based on the data set. Finally, a trainable Platt scaling is provided to convert the output of the string match to a posterior probability (same/different) for fusion with other entity resolution methods.

5.2 LiLAC

Entity resolution can also be performed based on content via author identification. The Lincoln Laboratory Author Classification (LiLAC) tool uses standard methods for author identification based on idiolect \[15\]. The basic strategy used in LiLAC is to represent an author by a document vector extracted from the content. Then, a one-vs-rest SVM is applied to train models for each of the authors. Finally, scoring an author model on a test set produces a matrix of similarity scores that can be used for author verification.
5.3 TweetE

As shown in Figure 3, entity resolution can be performed with graph methods. Our tool TweetE constructs a graph by designating both users (e.g., @twitter) and hashtags (e.g., #fashion) as vertices in the graph. For Twitter, edge types correspond to multiple categories: (1) user-to-user tweets, (2) user mentions of users or hashtags, (3) retweets, and (4) co-occurrence of hashtags or users. It is important to note that the graph construction technique can be applied to many other social media types, i.e., Reddit, Instagram, etc. For entity resolution, simple features based on common neighbors can be constructed, i.e., bag of common neighbors. Alternatively, community detection can be used; see [16, 17].

5.4 LLHash

Our final tool for entity resolution, LLHash, provides indexing of data via locality sensitive hashing and greedy agglomerative clustering [18]. Once pairwise similarity scores are generated, entity resolution can be cast as an unsupervised clustering problem. In [18], we demonstrate how the LLHash implementation of canopy clustering [19] can be used to reduce computational burden for the entity resolution problem.

6 Applications

6.1 Emotion Detection with pySLGR

An area of growing interest is the characterization and estimation of human emotion from multimedia data sources. As one task of the 2016 Audio-Visual Challenge (AVEC), we have considered the problem of arousal and valence estimation from various signals including speech, video, and physiological inputs. Arousal and valence are measures of emotion as a function of time given an input signal.

A system was rapidly developed for the speech component using pySLGR and other off the shelf components for a submission to the AVEC 2016 challenge [20]. The proposed pipeline consists of pySLGR signal processing and feature extraction, sparse modeling with the SPAMS toolkit, and regression using the SVM in scikit-learn. Because of the Python structure, integrating the components is quick and robust. Also, tuning the system with multiple iterations is straightforward. Performance is excellent on the DEV set with canonical correlation coefficient value of 0.796 for the arousal estimation.

6.2 Cross-Domain Entity Resolution

Associating entities (typically people and organizations) across multiple domains is a key problem in social media understanding. We have applied our tools to a large graph of users in Twitter and Instagram for entity resolution based on three categories: profile, content, and graph based. Specifically, LLString is used for matching usernames and full names across different user profiles. LiLAC is used for author identification based on the content of the posts and the writing style of the authors. Graph-based approach extracts user features based on graph structures and computes a similarity score between pairs of users across Twitter and Instagram graphs. TweetE is used to construct the Twitter and Instagram graph. Features based on common neighborhood and community membership are then computed and compared.

The performance of using the above-mentioned tools for entity resolution across Twitter and Instagram datasets is given in [17]. We have also combined the similarity scores from different features using random forest in scikit-learn. Performance is excellent with 5.74% equal error rate (EER) for profile-based features, 5.18% EER for combined profile and content based features, and 3.32% EER for the fusion of profile, content and graph based features.

7 Conclusions

We have described our LLTools framework for machine learning for human language processing. The tools in this framework span speech, text, and entity resolution methods. Our goals are to enable rapid development, wider adoption, and incorporation of human language processing into larger systems. Our tools are available at http://github.com/mitll.
8 Acknowledgements

This work would not have been possible without the direction and support of Chris White and Wade Shen, Program Managers of the DARPA XDATA and Memex programs. The authors would also like to acknowledge David Bringle, Douglas Gelbach, Brian Sandberg and the entire XDATA/Memex Performer community for their invaluable advice and support in the development of these tools.

References


