1 Implementation Track: Assessment Options (GA1)–(GA4)

For any of the options (GA1)–(GA4), you must

(i) implement the specific algorithm or system,

(ii) train it on the specified data,

(iii) evaluate it on the specified data,

(iv) create a written lab report/writeup that includes

  (a) a prose-based discussion of the algorithm;

  (b) a prose-based discussion of your implementation, including any hurdles you encountered (and how you addressed them);

  (c) a discussion of what tests you ran to ensure correctness;

  (d) results on both dev/validation and test splits; and

  (e) a summary of the approach's ability to do the task(s) you evaluated.

Be sure to cite appropriately and follow all academic honesty standards. Accounting for items like tables and graphs, an appropriate target length of this writeup is anywhere from 1 to 3 pages (though that is not a hard limit).

Within the Implementation Track, you will be primarly evaluated on the completeness and correctness of your implementation. However, the thoroughness and clarity of the writeup will be a sizeable (but non-majority) portion of your grade.

1.1 Milestones for (GA1)–(GA4)

Any of the Implementation Track options have three milestones prior to the final submission. All milestones must be met for full credit.

Milestone 1: Selection of Option This is due Monday October 9th by 11:59 PM. Select which of the options (GA1)–(GA4) you will do, and identify between 3 and 7 approved resources you think may be valuable to use in your work. Record your option selection, and upload a PDF list of the approved resources to the submission site, `https://www.csee.umbc.edu/courses/undergraduate/473/f23/submit`, selecting "GA: Milestone1." This list should not be your complete or final list of all resources you might use: it is meant as a starting point. You may also remove papers/resources from this list as you make progress.

Milestone 2: Initial draft of writeup, and git repo of your code with ≥ 3 non-trivial commits This is due Monday November 6th by 11:59 PM. You must submit two items: an initial version of your writeup, and a git repository of your implementation containing at least three, non-trivial commits (e.g., commits affecting more than just whitespace or comments). Your writeup does not need to include results, but it does need to include a prose-based discussion of the algorithm; a prose-based discussion of your implementation (or expected implementation), including any hurdles you are encountering; and a discussion of what tests you are running or will run to ensure correctness. Submit via the submission site, `https://www.csee.umbc.edu/courses/undergraduate/473/f23/submit`, selecting "GA: Milestone 2." You must turn in:

- an **ANONYMIZED** PDF of your writeup,
- a git repo for your code.

Your anonymized PDF will be provided to other students to review/provide feedback on (see Milestone 3); your code will not be shared. You may *optionally* provide a written description of what, if any, writing assistance you received (e.g., the GSA Writing Advisor).

Milestone 3: Feedback on Discussion This is due Friday November 17th by 11:59 PM. In this process, you will receive up to two other students' initial writeups; you must provide feedback on the breadth, depth, and clarity of exposition. You may also provide suggestions on hurdles that are described in the writeup(s). Reviewing forms and guides will be provided. To receive full credit for the reviews, you must provide constructive and civil reviews (a guide will be provided).

This feedback will be "double-blind:" as a reviewer, you will not know whose writeups you are reviewing, and as an author, you will not know who your reviewers are. This is why it is important for the Milestone 1 drafts to be anonymized. All paper-reviewer identities will be known to course staff.

Final Writeup and Full Code This is due Friday December 8th by 11:59 PM. This must be a **complete**, well-written writeup. These should be submitted to the submission site, `https://www.csee.umbc.edu/courses/undergraduate/473/f23/submit`, selecting "GA: Final Turn-in." You must turn in:

- a **NON-ANONYMIZED** PDF of the writeup,
- your code repo,
- a PDF document discussing the changes made, both as a result of the reviews/feedback and along with any unprompted changes.

As with the initial submission, you may also provide a written description of what, if any, writing assistance you received (e.g., the GSA Writing Advisor).

1.2 Specifics of Each of Implementation Track Option

(GA1): Implement a CRF For this option, implement a supervised linear chain conditional random field. This model was first described in Lafferty, J. D., McCallum, A., and Pereira, F. C. N. "Conditional random fields: Probabilistic models for segmenting and labeling sequence data" (ICML 2001), though you may also reference any of the textbooks for this class (e.g., chapter 7.5 of the Eisenstein book). You must implement this CRF from scratch (e.g., do not use any "CRF" library), and train and evaluate it on an appropriate sequence tagging task, such as POS tagging, named entity recognition (NER), or syntactic chunking. Your features must be static: that is, even if you use a neural method to compute their values, do not update the weights of the neural feature extractor when you're training your CRF. You can substitute a different gradient optimizer from the one they originally describe, e.g., L-BFGS, adagrad, Adam, etc.; this optimizer can be from a library like Pytorch The precise task is your choice: if doing POS tagging, you may use the UD data, though the instructor may need to provide separate data if you're interested in another task (like NER).

(GA2) Implement the inside-outside and CKY algorithms for probabilistic context free grammars. For probabilistic CKY, see chapter 14.2 of 3SLP. For inside-outside, while you may reference any resource in compliance with the "Y" resources in the beginnning, you may also reference Michael Collins's notes on the inside-outside algorithm: `http://www.cs.columbia.edu/~mcollins/io.pdf`. Consulting these notes will not be an academic integrity violation. Use the inside-outside algorithm to train in a semi-supervised manner (using the Penn Treebank sections 02-21 for supervised data, and the UD English EWT training sentences for unsupervised data), and evaluate on the Penn Treebank (dev evaluation on section

22, test evaluation on section 23). Obtain parses using your CKY implementation, and evaluate using the publicly available `evalb` script (`https://nlp.cs.nyu.edu/evalb/EVALB.tgz`).

(GA3) Implement and evaluate a neural arc-standard dependency parser. The algorithm is described in 3SLP or the Eisenstein book, and you may follow the broad outline given by Chen and Manning (2014, EMNLP), but note that you do not have to re-implement this work exactly. I *strongly* recommend that you run through the algorithm on paper, and really understand it, before you begin coding. Train and evaluate your system twice: the first time on English-EWT, the other is another language from UD of your choice. For this option, you may use existing layers in the toolkits Pytorch, Tensorflow, or Keras.

(GA4) Implement the Wall Street Journal/Penn Treebank perplexity results from Kim et al. (2016, AAAI: "Character-Aware Neural Language Models"). To do this, you will need to get the data from the instructor. In addition to re-creating the Penn treebank results, also evaluate this model on at least one non-English language from the UD data. For this option, you may use existing layers in the toolkits Pytorch, Tensorflow, or Keras.

Where to Start

You may analyze any papers read in class or as part of the assignments. **You are welcome and encouraged to come talk with course staff, either during office hours, Discord, or by appointment to discuss topics, advice on finding relevant papers, and the direction of your paper.**

Google Scholar is an easy way to find linked and cited papers. Another great resource is the ACL Anthology (`http://aclanthology.info/`), which archives papers by conferences (e.g., ACL, EACL, NAACL, NAACL), journals (CL, TACL), and workshops by year.[3] It also offers multiple custom searches: for example, searching "distributed representations" returns papers for crosslingual word representations (C12-1089), representations for relational patterns (P16-1215), and representations for semantic role labeling (D15-1295). Note that in NLP, conferences, and even workshops, are preferred to journals; conference reviewing can be just as, if not more intense and selective, as journal reviews. NLP conferences and workshops are almost always peer-reviewed and archival (meaning they are "finished" publications).

The AAAI digital library also offers an extensive listing of AI-based conferences and proceedings. Of particular relevance are the flagship AAAI, ICML (International Conference on Machine Learning), and KDD (Knowledge Discovery and Data Mining) proceedings. Papers from NeurIPS (Neural Information Processing Systems) often tend to the more theoretical, but with a decided focus on neural networks.

The following is a *very* small listing of potential starting papers:

1. Hinton (1986): "Learning Distributed Representations of Concepts"
2. Brown et al. (1992): "Class-based n-gram models of natural language"
3. Rosenfeld (1994): "Adaptive Statistical Language Modeling: A Maximum Entropy Approach", Chapters 5-8
4. Kneser and Ney (1995): "Improved backing-off for m-gram language modeling"
5. Bengio et al. (2003): "A Neural Probabilistic Language Model"
6. Ando and Zhang (2004): "A Framework for Learning Predictive Structures from Multiple Tasks and Unlabeled Data"
7. Rosenfeld (2004): "Two Decades of Statistical Language Modeling: Where Do We Go From Here?"
8. Mnih and Hinton (2008): "A Scalable Hierachical Distributed Language Model"
9. Graves (2013): "Generating Sequences with Recurrent Neural Networks"
10. Graves and Jaitly (2014): "Towards End-to-End Speech Recognition with Recurrent Neural Networks"
11. Devlin et al. (2014): "Fast and Robust Neural Network Joint Models for Statistical Machine Translation"

[3] Paper ids generally have the form *XYY-ZZZZ*, where X is a single letter identifier (P is the main ACL, D is for EMNLP, Q is for TACL, etc.), *YY* are the final two digits of the year ($2018 \rightarrow 18$), and *ZZZZ* is a per-proceedings identifier.