Knowledge of the Ancestors: Intelligent Ontology-aware Annotation of Biological Literature using Semantic Similarity

Pratik Devkota
Department of Computer Science
University of North Carolina at Greensboro, NC 27408
p_devkota@uncg.edu
Somya Mohanty
Department of Computer Science
University of North Carolina at Greensboro, NC 27408
sdmohant@uncg.edu
Prashanti Manda
Informatics and Analytics
University of North Carolina at Greensboro, NC 27408
p_manda@uncg.edu

Abstract

Natural language processing models have emerged as a solution to manual curation for fast and automated annotation of literature with ontology concepts. Deep learning architectures have particularly been employed for this task due to increased accuracy over traditional machine learning techniques. One of the greatest limitations in prior work is that the architectures do not use the ontology hierarchy while training or making predictions. These models treat ontology concepts as if they were independent entities while ignoring the semantics and relationships represented in the ontology. Here, we present deep learning architectures for ontology-aware models that use the ontology hierarchy for training and predicting ontology concepts for pieces of text. We explore the choice of three embeddings - CRAFT, GloVe, and ELMo to understand the impact on prediction performance. We evaluate our models using F1 and Jaccard semantic similarity and show that our ontology aware models can result in 2% - 10% (depending upon choice of embedding) improvements over a baseline model that doesn't use ontology hierarchies.

1 Introduction

Knowledge representation using ontologies has increased computational access to scientific data by making it possible to integrate, query, and run large-scale analyses on biological data. Computational analyses such as hypothesizing the genetic bases of evolutionary transitions to predicting gene functions to understanding rare human diseases are made possible by the availability of bio-ontology annotations [MBL^{+}15, GKM^{+}15]. These annotations are often the result of manual curation of literature - a slow and tedious process that is unscalable to the rapid pace of scientific publishing [DDI^{+}15].

Natural language processing (NLP) methods are one way to automate the manual curation process in an attempt to create techniques that can "read" and annotate biological literature with ontology concepts in

ICBO 2022, September 25-28, 2022, Ann Arbor, MI, USA EMAIL: p_devkota@uncg.edu (A. 1); sdmohant@uncg.edu (A. 2); p_manda@uncg.edu (A. 3) ORCID: 0000-0001-5161-0798 (A.1); 0000-0002-4253-5201 (A. 2); 0000-0002-7162-7770 (A. 3)

a fast, accurate, and scalable fashion. NLP techniques for automated ontology curation began with the use of traditional machine learning methods, rule based learning, lexical, and syntactic methods [BM18]. With limited accuracy, the focus has currently shifted to using deep learning for the task of automated ontology curation [MBM18, MSM20a].

Deep learning architectures have gained popularity in several areas such as image processing, audio analysis, etc, but especially so for text analysis. In ontology-based NLP tasks, deep learning models have been applied to named entity recognition (NER), named entity normalization (NEN), and relation extraction [SJC$^+$22, YRSE20, XLS20]. NER focuses on identifying entities from unstructured text such as scientific literature while NEN approaches focus on linking the entities to unique concepts such as terms in an ontology. Relation extraction tasks focus on identifying relations between the identified ontology concepts and enable automated creation of structured hierarchies. Several deep learning models such as Long Short Term Memory (LSTM), Gated Recurrent Units (GRU), Recurrent Neural Networks (RNN), etc. have been employed for NER and NEN tasks with varying levels of success. In prior work, we evaluated several existing deep learning models and developed new architectures based on Gated Recurrent Units [MBM18, MSM20a]. While the models presented were successful at performing NER of ontology concepts from text, they lacked awareness of the underlying ontology hierarchy. It is critical for ontology-based information retrieval systems to know the ontology structure and relationships so that they can make intelligent predictions for concepts that take into account patterns learned from the training data as well as semantics embedded in the ontology.

The key difference in traditional information retrieval vs. ontology-based information retrieval is the possibility of partial success. Traditional information retrieval systems are evaluated based on whether the target information is retrieved (success) or not (failure). In contrast, ontology-based information retrieval systems are evaluated based on three possibilities: accurate retrieval (success), inaccurate retrieval (failure), or partially accurate retrieval (partial success).

If a model accurately predicts the ontology concept from the gold standard data, it is counted as success and is scored a 1. If the model fails to predict any ontology concept, it is counted as failure and is scored a 0. If the model retrieves an alternative ontology concept from the one in the gold standard (partial success), the model is scored depending on how semantically similar the retrieved concept is to the true concept. Intuitively, the most semantically similar concept for the true concept would lie in the near vicinity (such as a sibling, or a parent). Thus, it is important for the model to be aware of the ontology hierarchy to be able to make intelligent predictions in cases when it misses the actual concept. The goal of intelligent concept annotation presented in this study is to maximize accurate retrieval rates and subsequently maximize partial accuracy in cases where complete accuracy is not achieved thereby improving overall accuracy.

This nuance of ontology-based concept retrieval means that the first goal of these models is to predict the true concept. If the first goal is not achieved, the model should aim to predict the most semantically similar concept to the truth so as to maximize the partial accuracy score. The ontology hierarchy contains valuable information regarding the semantics embedded in the ontology. This crucial information has been ignored in past work resulting in ontology recognition models that are not fully aware of the ontology semantics. Here, we present intelligent deep learning architectures that are ontology-aware and use the hierarchies embedded in the ontology to improve concept prediction accuracy.

Gold standard corpora annotated with ontology concepts are necessary for training and evaluating natural language processing models. Training deep learning models particularly requires high quality training datasets. The Colorado Richly Annotated Full Text Corpus (CRAFT) [BEE$^+$12] is a widely used training resource for automated annotation approaches. The current version of the CRAFT corpus (v4.0.1) provides annotations for 97 biological/biomedical articles with concepts from 10 ontologies including the Gene Ontology (GO).

2 Related Work

The majority of work in the areas of natural language processing techniques for ontology-based concept recognition is aimed at predicting GO concepts. This is understandable since the GO is one of the most widely used bio-ontologies.

In a previous study [MBM18], we presented a deep learning architecture that used multiple GRUs with a character and word based input. The model was compared to seven models from existing work using the CRAFT corpus as a gold standard. Results showed that our GRU-based multi-pipeline model outperformed prior models such as LSTMs, RNNs, etc. This work was limited to predicting unigram annotations and did not take into account the rich semantic information in ontology hierarchies. Subsequent work [MSM20a] from our group improved on this by expanding the types of annotations predicted and by incorporating semantics from ontology subsumption into the prediction. Surprisingly, we found that GRU based models consistently outperformed the commonly used LSTM based architectures. We developed a rudimentary approach to incorporate ontology hierarchy into prediction by generalizing the training data to different levels in the ontology using subsumption reasoning and running independent models on the different datasets. The approach only resulted in a modest improvement in performance [MSM20a].

Most recent publications in this area have separated the ontology annotation task to two sub-tasks - 1) span detection: detecting the part of text that corresponds to an ontology concept, and 2) concept normalization: identifying the ontology concept most appropriate for the identified piece of text [BHB+21, HBHH19]. Using the CRAFT corpus as a training set, the study reports that Bidirectional encoder representations from transformers (BERT) resulted in the best performance (0.81 F1) for the span detection sub-task.

Using similar approaches, [FCR19] compared the performance of an LSTM model with BERT. This study divided the ontology annotation task into span detection and named entity normalization (NEN). This step enables the models to predict concepts that might not be seen in the training data. Note that this system currently cannot handle sophisticated annotation formats in the CRAFT corpus.

The application of deep learning architectures such as RNNs and LSTMs has also been explored for the task of relationship extraction between ontology concepts [SLC21]. Sousa et. al. discuss neural network models to perform text mining tasks on data structures such as ontologies. Biological annotation tools such as Textpresso [MKSA04] conduct automatic curation by automatically extracting ontological entities and the relations between them. Similarly, other studies have facilitated relation extraction by creating gold standard datasets such as BioRel [XLS20]. BioRel is a large-scale dataset designed specifically for relation extraction problems using Unified Medical Language System as the source. Self attentive networks have also been found to be promising and have been applied for identifying drug-drug interactions, protein-protein interactions, as well as for identifying relations between medical concepts [YRSE20].

3 Methods

3.1 Training Dataset

This study used version v4.0.1 (`https://github.com/UCDenver-ccp/CRAFT/releases/tag/v4.0.1`) of The Colorado Richly Annotated Full Text Corpus (CRAFT) [BEE+12], a manually annotated corpus containing 97 articles each of which is annotated to 10 ontologies. We selected GO annotations from the CRAFT corpus as our training and testing set because the largest number of annotations in CRAFT are made using the GO.

3.2 Data Preprocessing

The first step is to preprocess each annotation in the CRAFT into a format that can be used by the deep learning models. The following preprocessing steps are performed to translate annotations from the CRAFT corpus to the desired input formats.

3.2.1 Sentence Segmentation and Tokenization

Annotations in the CRAFT corpus are recorded via character index spans that indicate the beginning and end of an annotation. First, the segmenter splits the text into sentences by accounting for sentence end marks (such as periods, exclamation, question marks, etc.) and then uses a tokenizer to split the sentences into individual words (or tokens) by accounting for word boundaries (such as space, hyphen, tab, etc.).

3.2.2 IOB Tagging

Each extracted word/token is mapped to a GO term or an *out-of-concept* annotation. Each token is mapped to one of the following three categories ($\mathcal{T}$): 1) `GO` to indicate an annotation, `'O'` for a non-annotation (out-of-concept), and `'EOS'` to indicate the end of sentence.

In some cases, a sequence of words/tokens is annotated to a GO term. In these cases, we use the IOB (Inside, Outside, Beginning) [RM95] standard. The IOB format uses three prefixes to tag tokens in a sentence: 1) `'B-GO'` is used to specify the beginning of the annotation, 2) `'I-GO'` is used to map the tokens following the beginning of annotation till the end, and 3) `'O'` is used to map tokens that don't correspond to a GO term.

3.2.3 Annotation Formats

Sentences in the CRAFT corpus are annotated following a set of annotation formats and guidelines as detailed in `https://github.com/UCDenver-ccp/CRAFT/tree/master/concept-annotation`. Below, we describe how sentences that contain annotations in different formats are represented in the IOB format.

- **No annotations**: Some sentences in an article might not contain any annotations. In this case, all tokens are represented by `'O'` tags except the ending character which is represented by `'EOS'` tag.

- **Disjoint annotations:** A sentence might contain one or more annotations that don't overlap in terms of annotation span. In this case, all tokens not corresponding to an annotation are tagged with **O** tags. The end of sentence character is represented by **EOS** tag. Tokens that mark the beginning of an annotation are marked with a **B-GO:term** followed by **I-GO:term** to represent subsequent tokens corresponding to an annotated phrase.

- **Overlapping annotations:** Here we show an example of a sentence containing annotations with overlapping spans. In this case, a phrase (sequence of words/tokens) is annotated to a GO concept, and a word or a sub-phrase within the original phrase is annotated to a different GO concept.

  Sentence: *"Having excluded a direct role in **vesicle formation** and membrane fusion, annexin A7 might act by its property as Ca2+-binding protein"*

  Annotations: {*'vesicle'* — `GO:0031982`; 'vesicle formation' — `GO:0006900`}

  In these instances, we make n copies of the sentence where n is the number of different annotations. Each copy contains a modified sentence that represents the text needed to convey one of the annotations. The above example is represented as two sentences with each sentence representing one of the two annotations.

  Sentence 1: *"Having excluded a direct role in **vesicle** and membrane fusion, annexin A7 might act by its property as Ca2+-binding protein"*

  Annotations: {*'vesicle'* — `GO:0031982`}

  Sentence 2: *"Having excluded a direct role in **vesicle formation** and membrane fusion, annexin A7 might act by its property as Ca2+-binding protein"*

  Annotations: {*'vesicle formation'* – `GO:0006900`}

  If a sentence contains a case of overlapping annotations and other disjoint annotations (non-overlapping annotations), we create sentences that capture the different variations of the overlapping annotations while keeping the disjoint annotations common.

- **Multiple overlapping annotations**: Sentences can also have more than one phrase with sub-annotations. In such a case, where there exist m phrases with $n_1, n_2, \cdots, n_m$ overlapping sub-phrases, there will $n_1 \times n_2 \times ... \times n_m$ copies with all possible combinations of sub-phrase mappings.

  Sentence: *"Having excluded a direct role in **vesicle formation** and **membrane fusion**, annexin A7 might act by its property as Ca2+-binding protein."*

  Annotations: {*'vesicle'* — `GO:0031982`; *'vesicle formation'* — `GO:0006900`; *'membrane'* — `GO:0016020`; *'membrane fusion'* — `GO:0061025`}

  In this example, we have two instances of overlapping annotations with two sub-phrase annotations each. This sentence would be transformed to four sentences that each represents a unique combination of annotations.

  Sentence 1: *"Having excluded a direct role in **vesicle** and **membrane**, annexin A7 might act by its property as Ca2+-binding protein."*

  Annotations: {*'vesicle'* — `GO:0031982`; *'membrane'* — `GO:0016020`}

  Sentence 2: *"Having excluded a direct role in **vesicle formation** and **membrane**, annexin A7 might act by its property as Ca2+-binding protein."*

  Annotations: {*'vesicle formation'* — `GO:0006900`; *'membrane'* — `GO:0016020`}

  Sentence 3: *"Having excluded a direct role in **vesicle** and **membrane fusion**, annexin A7 might act by its property as Ca2+-binding protein."*

  Annotations: {*'vesicle'* — `GO:0031982`; *'membrane fusion'* — `GO:0061025`}

  Sentence 4: *"Having excluded a direct role in **vesicle formation** and **membrane fusion**, annexin A7 might act by its property as Ca2+-binding protein."*

  Annotations: {*'vesicle formation'* — `GO:0006900`; *'membrane fusion'* — `GO:0061025`}

- **Discontinuous annotations**: Some sentences in the CRAFT corpus contain discontinuous annotations where non-consecutive words/tokens are annotated to a single concept, while tokens between them are not.

  Sentence: *"Because the F7 is the most severely affected allele, it is possible that the difference between the heart and kidney levels is due to a developmental delay in **v**/p **formation**."*

  Annotations: *"v formation"* — `GO:0097084`

  Here we see *"v formation"* is annotated to `GO:0097084`, whereas *"/p"* is not. In such a case we represent the sentence by removing the tokens/words which were not annotated (*"/p"*). This is done to represent the continuous span of the phrase to GO term mapping.

Transformed Sentence: *"Because the F7 is the most severely affected allele, it is possible that the difference between the heart and kidney levels is due to a developmental delay in **v formation**."*

3.2.4 POS Tagging and Token Encoding

Following the tokenization and IOB tagging, we enrich training data with parts-of-speech (POS) information and a compressed character representation. POS tagging looks at the contextual information of the word based on the words surrounding it in a sentence or a phrase. Here, we tag each token with 15 parts of speech tags — adjective, adposition (such as - in, to, during), adverb, auxiliary (such as - is, has done, will do, should do), conjunction, coordinating conjunction, determiner, interjection, noun, numeral, particle, pronoun, proper noun, punctuation, subordinating conjunction, symbol, verb, other, space.

While the POS tagging looks at the word level representation of the context, we also represent character level nuances of a token using character encodings. These encodings represent upper-case and lower case characters with 'C' or 'c' respectively. Numbers are represented using an 'N' and punctuation (such as commas, periods, and dashes) are retained in the encoding.

3.3 Deep Learning Architecture

After all the preprocessing steps described above are complete, we develop multidimensional vectors for each sentence of the articles. Our deep learning architecture (Figure 1) consists of three key components — 1) input pipelines; 2) embedding/latent representations; and 3) a deep learning model.

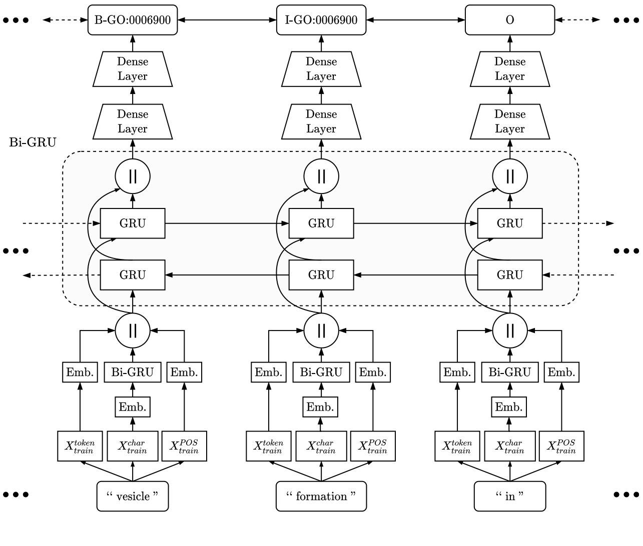

Figure 1: Workings of a Gated Recurrent Unit based model with an example input sequence

3.3.1 Input Pipelines

The neural architectures in this study are designed to use fixed size inputs. We explored the frequency distribution of words in the CRAFT dataset to determine the size required for the architectures (71 is 3-SD). Each sentence was transformed to a size of 71 words based on the third standard deviation of the word frequency distribution. Sentences with lower number of words are padded with the token `<PAD>` and ones with higher number of tokens are truncated to a length of 71.

We provide three inputs for each word in a sentence - 1) token (X^{token}_{train}), 2) character sequence (X^{char}_{train}), 3) parts-of-speech (X^{POS}_{train}).

The token (X^{token}_{train}) input, is a sequential tensor consisting of 71 tokens, where each token is represented with a high dimensional one hot encoded vector (for 34,164 unique words/tokens present within our corpus vocabulary). Similarly, the character sequence (X^{char}_{train}) is also a sequential tensor consisting of character sequences present in a word/token. Next we provide POS tags that indicate the type of words in a sentence (X^{POS}_{train}).

3.3.2 Embedding/Latent Representations

Our architecture utilizes embeddings to provide a compressed latent space representation for very high dimensional input components. For example, the one hot vectorization of an individual word has a dimensionality of 34,166. In order to represent them succinctly and with contextual representation, we evaluated three different approaches for embeddings (shown as Emb. in Figure 1) - 1) CRAFT 2) Global Vectors for Word Representation(GloVe) , 3) Embeddings from Language Models (ELMo).

The supervised embedding is a bottleneck layer which learns to map the one hot encoded input into a smaller dimensional representation. The resulting embedding learns the mapping of the IOB tags to the tokens of the sentences. The layer is used with token inputs (X^{token}_{train}), character sequences (X^{char}_{train}), and character representation (X^{POS}_{train}), each of which have very high dimensionality in their original vectors.

We also evaluate GloVe [PSM14] and ELMo [PNI^{+}18] pretrained embeddings for the X^{token}_{train} input. GloVe uses word co-occurrence statistics to learn the embeddings. In comparison, the embeddings in ELMo are learned via a bidirectional language model where the sequence of the words are also taken into account. While ELMo and GloVe are large pretrained (on external corpus) embedding models, we also evaluate the models performance using the CRAFT embedding. Here we use a embedding layer which is cotrained during the model training, where each word is mapped to an embedding vector which is a continuous vector of 100 dimensions. Initially the embedding values of the layer are randomly initialized, but are then optimized for word lookup during the model training for the objective of ontology annotation.

3.3.3 Deep learning model

Our architecture utilizes a bi-directional gated recurrent model (Bi-GRU). Traditional Recurrent Neural Networks suffer from short term memory that isn't ideal when training using long sentences. In long sequences of words, these models will forget patterns from the beginning of the sentence as they move further into the sentence. The gradients computed during back propagation gradually shrink causing small gradients from earlier parts of the inputs that don't contribute to learning. This problem known as the vanishing gradient problem makes it ineffective for RNNs to learn from long pieces of text.

Gated Recurrent Units [CVMG$^+$14] employ two gates a reset gate and an update gate to address the vanishing gradient problem. The reset gate handles short term memory while the update gate handles long term memory. The reset gate allows the user to specify how much of the prior states to remember. The combination of these two gates enables GRUs to retain information longer without diluting their impact to learning.

The choice of GRUs for our architecture is not arbitrary. We conducted an evaluation of RNNS, LSTMs, and GRUs in prior work [MSM20b, MBM18], and found that the GRU based architecture performed the best.

Figure 1 shows a snapshot of the model architecture in the context of training and inference of a sample set of tokens. Here we show the training/inference on a sequence of tokens "vesicle", "formation", and "in" (which are parts of a sentence) as it is evaluated by the network. Each token is preprocessed to obtain the representative tensors – X^{token}_{train}, X^{char}_{train}, X^{POS}_{train} which are passed through embedding layers, where the embedding of X^{token}_{train} can be a complete pretrained architecture such as GloVe or ELMo. The embedding of X^{char}_{train} is also passed via a Bi-directional GRU (Bi-GRU) layer. All of the resulting values are concatenated to be processed via the main Bi-GRU layer. The bi-directionality allows the architecture to learn the preceding and succeeding sequence patterns within the sequence tokens in a sentence. The states of both the GRU layers are then concatenated to provided to the final dense layer, which is the softmax classifier of the architecture, and predicts the associated IOB tags for the input tokens. Here we select the tag with the highest probability for each of the tokens.

Architecture hyper-parameters, which include supervised embedding shape ({20, 50, 100, 150, 200}), dropout ({01, .2, .3, .5, .7}), number of epochs ({50, 100, 200, 300}), and class weighting, were evaluated using a grid search approach. We used Adam [KB17] as our optimiser for all of the experiments with a default learning rate of 0.0001.

We also compare our approach to large scale masked language models such as BERT. Bidirectional Encoder Representations from Transformers (BERT) [DCLT18] is a popular attention model developed by Google. BERT has rapidly become the state of the art in several applications, especially those involving text processing. We compared the best model from our experiments above with BERT.

3.4 Target Vector Representation

Target labels to be predicted are typically provided as a one-hot encoded vector where the size of the vector equals the number of output labels. In our case, the output labels correspond to the set of all GO terms. Typically, the value of the GO term to be predicted is set to a 1 and the value of all other GO terms is set to 0. This approach of representing the target labels, however, does not allow the model to learn the ontology hierarchy nor does it allow for semantically similar partial predictions.

In this study, we use Jaccard semantic similarity scores as values in the label vector. The value of the GO term to be predicted is set to 1 and the value of all other GO terms in the vector is set to the Jaccard similarity score between that term and the GO term to be predicted. This representation allows the model to identify the target GO term followed by "similar" GO terms that are partially accurate predictions. This output label representation also helps the model optimize the weights to target more than one prediction label. We also add in a weighting factor β to modulate the effect of Jaccard similarity score on the target vector. So for each output tag, the representation is as follows:

$$Y = \begin{cases} l = [1], & \text{if } \mathcal{T} == \widehat{\mathcal{T}} \\ l = [\beta * \mathcal{J}_{sim}(\mathcal{T}, \widehat{\mathcal{T}})], & \text{if } \mathcal{T} \neq \widehat{\mathcal{T}} \text{ \& } \mathcal{T} \neq \mathtt{0} \end{cases} \tag{1}$$

where, Y is the final target vector, l is the label for the word, $\mathcal{T}$ is the ground truth tag, $\widehat{\mathcal{T}}$ is the predicted tag, β is the Jaccard weight, and $\mathcal{J}_{sim}$ is the Jaccard similarity between $\mathcal{T}$ and $\widehat{\mathcal{T}}$. The target vector Y is computed by comparing the true tag ($\mathcal{T}$) with the possible tags ($\widehat{\mathcal{T}}$), where if the $\mathcal{T} == \widehat{\mathcal{T}} == O$ (no annotation) OR a GO annotation then the value is set to 1. Else, if the ground truth tag ($\mathcal{T}$) is a GO term, then we calculate its Jaccard similarity to all possible

GO terms and create the target vector by weighting it with β. We evaluate β values between {0, .25, .5, 1}, where a β value 0 indicates the traditional one-hot vectorization (baseline in results) and a β value 1 indicates the full Jaccard score taken into account.

The Jaccard similarity ($\mathcal{J}_{sim}$) of the ground truth concept $\mathcal{T}$ and a predicted concept $\widehat{\mathcal{T}}$ [PFF+09] is calculated as:

$$\mathcal{J}_{sim}(\mathcal{T}, \widehat{\mathcal{T}}) = \frac{|S(\mathcal{T}) \cap S(\widehat{\mathcal{T}})|}{|S(\mathcal{T}) \cup S(\widehat{\mathcal{T}})|} \tag{2}$$

where, $S(\mathcal{T})$ is the set of ontology subsumers of $\mathcal{T}$. Specifically, $\mathcal{J}_{sim}$ of two concepts (A, B) in an ontology is defined as the ratio of the number of concepts in the intersection of their subsumers over the number of concepts in their union of their subsumers [PFF+09].

3.5 Performance Evaluation Metrics

The performance of each experiment is evaluated using a modified F1 score. The model is tasked with predicting non-annotations (indicated by an 'O' tag) or annotations (indicated by a 'GO' tag). Since the majority of tags in the training corpus are non-annotations, the model predicts them with great accuracy. In order to avoid biasing the F1 score, we omit accurate predictions of 'O' tags from the calculation to report a relatively conservative F1 score.

F1 quantifies whether the model's prediction matches the actual annotation exactly. However, ontology-based prediction systems need to be evaluated while accommodating partially accurate predictions. For example, a model might not retrieve the exact ontology concept as the gold standard but a related concept (sub-class or super-class) achieving partial accuracy. Semantic similarity metrics [PFF+09] designed to measure different degrees of similarity between ontology concepts can be leveraged to measure the similarity between the predicted concept and the actual annotation to quantify the partial prediction accuracy. Here, we use Jaccard similarity which measures the ontological distance between two concepts, to access the models performance for partial similarity between the predicted tags and the actual ground truth.

3.6 Top two predictions

The model makes predictions for each GO instance in the test set. These predictions are expressed as a probability vector where each potential GO term is assigned a probability. When evaluating the performance of the model, we typically pick the GO term with the highest probability and use that as the model's prediction. However, it is typical in NLP evaluations to consider the top 2, 5, and even 10% of probabilities in evaluating the performance of the model [BHA+21, JK19, PSY+18]. In one of our prior works, we showed that using the top 2 probabilities can result in a substantial increase in accuracy [MSM20a]. Here, we use the same practice of considering predictions with the top 2 probabilities to evaluate our model's performance.

4 Results and Discussion

The CRAFT v4.0.1 dataset contains 18,689 annotations pertaining to 974 concepts from the three GO sub-ontologies across 97 articles. The majority of these concepts belong to Biological Process followed by Molecular Function, and Cellular Component.

First, we look at a comparison of the three embeddings tested - CRAFT, GloVe, and ELMo. We establish a baseline performance by training the model with a binary target vector and not using Jaccard similarity scores. This baseline can help understand the performance improvements resulting from training the model with semantic similarity scores. Row 1 of Table 2 shows the baseline F1 and Jaccard similarity with the three embeddings. We see that ELMo results in the highest F1 (0.79) and the highest Jaccard score (0.82). Considering the top 2 predictions increased the F1 to 0.86 and the Jaccard score to 0.90.

We tested the prediction performance with four different settings of the weight parameter β (0.25, 0.5, 0.75, and 1). We found that the best performance was obtained with a weight of 0.5 (Row 2, Table 2). In both the baseline and the ontology aware model, the ELMo embedding outperforms the other two embeddings across all metrics.

The impact of training the model with the ontology hierarchy is starkly noticeable in both F1 and Jaccard across all three embeddings. The highest improvement is observed for CRAFT embeddings (8% F1, 10% Jaccard) followed by GloVe (6% F1, 8% Jaccard). ELMo showed modest improvements between the baseline and the ontology aware model (2% F1 and Jaccard). These results suggest that intelligent models that are trained with the ontology hierarchy make more accurate predictions as compared to those that are trained just on the target ontology concept.

We present a few examples of instances where the model's predictions match the ontology term in the gold standard as well as instances where the model generates false negatives or partially accurate annotations (Table 1).

We also compare the transformer based BERT model which has been shown to perform state of art results in a large of named entity recognition and other

Phrase	Gold standard annotation	Model's annotation
Accurate annotation prediction		
endogenous **intracellular**	GO:0005622	GO:0005622
cell divisions before developmental	GO:0051301	GO:0051301
Partially accurate annotation prediction		
proper **rhabdomere morphogenesis**	GO:0061541	GO:0031069
auto **regulation of blood flow**	GO:1903522	GO:0008217
Inaccurate annotation prediction		
polysomal -associated RNA-binding protein	GO:0005844	'O'
protein degradation	GO:0030163	'O'

Table 1: Examples of accurate, partially accurate, and inaccurate annotation predictions.

NLP tasks. Results show that our ontology aware model outperforms BERT in both F1 and Jaccard by 5%. The baseline model that shows a lower performance than the ontology aware model also outperforms BERT in our test.

It is interesting to see that a generic ELMo embedding performs better than a domain specific embedding such as CRAFT. However, it is to be noted that the ELMo and GloVe embeddings are pretrained on large corpora. They are computationally more expensive to inference during model training and even more expensive to develop. In contrast, CRAFT utilizes a simple embedding layer with significantly lower number of parameters making it more accessible for scientists.

Prior approaches in the area of automated ontology annotation treated ontological concepts as a binary outcome. In our approach, the Jaccard similarity target vector teaches the model to understand the latent semantic relationships between the GO concepts.

5 Conclusion

This work introduced approaches for incorporating the ontology hierarchy to make more accurate concept predictions for text. We show that our intelligent ontology-aware model results in higher annotation accuracy over a naive baseline model. This work paves the way for more sophisticated approaches for enabling deep learning architectures to gain understanding of the ontology space and semantics.

6 Data and Code Availability

The data used in this work is publicly available at https://github.com/UCDenver-ccp/CRAFT/releases/tag/v4.0.1. The code used to generate the results can be found at https://github.com/prashanti/intelligentannotation. The code is archived on Zenodo (doi: 10.5281/zenodo.6964353).

Model	Embed.	F1	Jaccard	Top two F1	Top two Jaccard
Baseline	CRAFT	0.74	0.75	0.82	0.86
	GloVe	0.75	0.76	0.832	0.87
	ELMo	0.79	0.82	0.86	0.90
Ontology aware model	CRAFT	0.80	0.83	0.86	0.91
	GloVe	0.79	0.82	0.86	0.90
	ELMo	0.81	0.84	0.87	0.92

Table 2: Comparison of model performance at the baseline and after using the ontology hierarchy for training. Note: F1 scores were modified by omitting accurate predictions of non-annotations (indicated by 'O') for a conservative estimate of performance on annotations only.

Model	F1	Jaccard
BERT	0.77	0.80
Ontology aware model (ELMo)	0.81	0.84

Table 3: Performance comparison between our best model and BERT. Note: F1 scores were modified by omitting accurate predictions of non-annotations (indicated by 'O') for a conservative estimate of performance on annotations only.

7 Funding

This work is funded by a CAREER grant from the Division of Biological Infrastructure at the National Science Foundation (#1942727).

References

[BEE$^+$12] Michael Bada, Miriam Eckert, Donald Evans, Kristin Garcia, Krista Shipley, Dmitry Sitnikov, William A. Baumgartner, K. Bretonnel Cohen, Karin Verspoor, Judith A. Blake, and Lawrence E. Hunter. https://doi.org/10.1186/1471-2105-13-161Concept annotation in the CRAFT corpus. *BMC Bioinformatics*, 13(1):161, Jul 2012.

[BHA+21] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[BHB+21] Mayla R Boguslav, Negacy D Hailu, Michael Bada, William A Baumgartner, and Lawrence E Hunter. Concept recognition as a machine translation problem. *BMC bioinformatics*, 22(1):1–39, 2021.

[BM18] Lucas Beasley and Prashanti Manda. Comparison of natural language processing tools for automatic gene ontology annotation of scientific literature. *Proceedings of the International Conference on Biomedical Ontology*, 2018.

[CVMG+14] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[DCLT18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[DDI+15] Wasila Dahdul, T Alexander Dececchi, Nizar Ibrahim, Hilmar Lapp, and Paula Mabee. Moving the mountain: analysis of the effort required to transform comparative anatomy into computable anatomy. *Database*, 2015, 2015.

[FCR19] Lenz Furrer, Joseph Cornelius, and Fabio Rinaldi. Uzh@ craft-st: a sequence-labeling approach to concept recognition. In *Proceedings of The 5th Workshop on BioNLP Open Shared Tasks*, pages 185–195, 2019.

[GKM+15] Tudor Groza, Sebastian Köhler, Dawid Moldenhauer, Nicole Vasilevsky, Gareth Baynam, Tomasz Zemojtel, Lynn Marie Schriml, Warren Alden Kibbe, Paul N Schofield, Tim Beck, et al. The human phenotype ontology: semantic unification of common and rare disease. *The American Journal of Human Genetics*, 97(1):111–124, 2015.

[HBHH19] Negacy D Hailu, Michael Bada, Asmelash Teka Hadgu, and Lawrence E Hunter. Biomedical concept recognition using deep neural sequence models. *bioRxiv*, page 530337, 2019.

[JK19] Justin M Johnson and Taghi M Khoshgoftaar. Survey on deep learning with class imbalance. *Journal of Big Data*, 6(1):1–54, 2019.

[KB17] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[MBL+15] Prashanti Manda, James P Balhoff, Hilmar Lapp, Paula Mabee, and Todd J Vision. Using the phenoscape knowledgebase to relate genetic perturbations to phenotypic evolution. *genesis*, 53(8):561–571, 2015.

[MBM18] Prashanti Manda, Lucas Beasley, and Somya Mohanty. Taking a dive: Experiments in deep learning for automatic ontology-based annotation of scientific literature. *Proceedings of the International Conference on Biomedical Ontology*, 2018.

[MKSA04] Hans-Michael Müller, Eimear E Kenny, Paul W Sternberg, and Michael Ashburner. Textpresso: an ontology-based information retrieval and extraction system for biological literature. *PLoS biology*, 2(11):e309, 2004.

[MSM20a] Prashanti Manda, Saed SayedAhmed, and Somya D Mohanty. Automated ontology-based annotation of scientific literature using deep learning. In *Proceedings of The International Workshop on Semantic Big Data*, pages 1–6, 2020.

[MSM20b] Prashanti Manda, Saed SayedAhmed, and Somya D. Mohanty. Automated ontology-based annotation of scientific literature using deep learning. In *Proceedings of The International Workshop on Semantic Big Data*, SBD '20, New York, NY, USA, 2020. Association for Computing Machinery.

[PFF+09] Catia Pesquita, Daniel Faria, Andre O Falcao, Phillip Lord, and Francisco M Couto. Semantic similarity in biomedical ontologies. *PLoS computational biology*, 5(7), 2009.

[PNI$^+$18] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *CoRR*, abs/1802.05365, 2018.

[PSM14] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[PSY$^+$18] Samira Pouyanfar, Saad Sadiq, Yilin Yan, Haiman Tian, Yudong Tao, Maria Presa Reyes, Mei-Ling Shyu, Shu-Ching Chen, and Sundaraja S Iyengar. A survey on deep learning: Algorithms, techniques, and applications. *ACM Computing Surveys (CSUR)*, 51(5):1–36, 2018.

[RM95] Lance Ramshaw and Mitch Marcus. Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora*, page 6, 1995.

[SJC$^+$22] Mujeen Sung, Minbyul Jeong, Yonghwa Choi, Donghyeon Kim, Jinhyuk Lee, and Jaewoo Kang. Bern2: an advanced neural biomedical named entity recognition and normalization tool. *arXiv preprint arXiv:2201.02080*, 2022.

[SLC21] Diana Sousa, Andre Lamurias, and Francisco M Couto. Using neural networks for relation extraction from biomedical literature. In *Artificial Neural Networks*, pages 289–305. Springer, 2021.

[XLS20] Rui Xing, Jie Luo, and Tengwei Song. Biorel: towards large-scale biomedical relation extraction. *BMC bioinformatics*, 21(16):1–13, 2020.

[YRSE20] Shweta Yadav, Srivastsa Ramesh, Sriparna Saha, and Asif Ekbal. Relation extraction from biomedical and clinical text: Unified multitask learning framework. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2020.