Cognitive States:

Belief State Inference via Deep Learning

Tyler Osborne

Advised by Amittai Aviram

CSCI4961 Honors Thesis

Department of Computer Science, Boston College

Chestnut Hill, MA 02467

12 May 2023

Table of Contents

0. Dedication

This thesis is foremost dedicated to my parents, who put up with my stress and grumpiness as I worked to finish up against the deadline. Furthermore, this thesis is dedicated to my professors and mentors, especially Prof. Aviram, without whom I am sure I would never have embarked on this thesis in the first place.

I. Abstract

The Cognitive States project is an ongoing investigation of how well a pre-trained deep learning model, fine-tuned with various corpora of annotated texts, can infer belief states. Overall, the goal of this project is to make incremental progress towards more advanced belief state and sentiment detection capabilities. Previous research has focused on achieving state-of-the-art F1 results on classification tasks as well as end-to-end generative tasks defined on two corpora annotated for belief, sentiment, or both, Factbank and MPQA, using two models, BERT and Flan-T5 (Murzaku et al). We use the same models to define similar tasks on the Language Understanding (LU) corpus, in order to corroborate insights gained from previous work. Furthermore, we present a novel database representation for fine-tuning data, allowing for the unification of Factbank, MPQA, LU, and additional annotation-based belief/sentiment corpora into a single dataset for seamless use in multi-task learning contexts, requiring unifying data transformations such as converting unigram head words to n-gram spans. Our results for LU's majority class align with those of Murzaku et al. on all tasks, whereas our approaches performed less well on minority classes. Plans to improve minority performance include leveraging a few-shot approach or generating synthetic data by swapping out words in existing examples with close synonyms.

II. Introduction

In linguistics, belief state is the degree to which a speaker believes in the truthfulness of their utterance. When a speaker introduces new information, they communicate more than what they explicitly say; among other things, they also convey how sure they are that what they are saying is true. Of course, a speaker is unlikely to say, for example, "John said Mary is coming to dinner with sixty-five percent confidence." Rather, a listener would piece together John's confidence in Mary's attendance at dinner through contextual features contained in the surrounding text, thus only requiring the author to say something like, "John said that Mary might come to dinner," to convey essentially the same information.

There is also the issue of the sentence's author, who is not explicitly mentioned in either of the given sentences, and their level of belief regarding the truthfulness of John's utterance. Two terms must be introduced before we continue: source and target. A source is a person in the text who expresses a belief (including the author), and a target is the chunk of text indicating the expression of the source's belief. John and the author of the sentence are therefore two distinct *sources* of belief expression. Their respective claims, which for John (in the first example) is that Mary will come to dinner and for the author that John said Mary is coming to dinner, are referred to as *targets*, which can be single tokens or an n-gram. Sources, targets, and the sentences in which they appear are all housed in annotated corpora where a corpus is simply a collection of text.

For each corpus, humans carefully analyze the text and manually annotate sources and targets as they appear in order to link source-target pairs to belief labels (i.e., a value describing a source's level of belief; the sets of possible labels differ between corpora), creating a source-target-*label* triple. Note that sources and targets are identified by character *offsets*

indicating at which characters in the sentence the target begins and ends. Throughout our research, we have used such source-target-label triples as training data to fine-tune pre-trained deep learning models that will be described later, where either the source and target are given and the label is predicted by the model (classification task), or the sentence alone is given and tuples of the form *(source, target, factuality)* are outputted by the model (end-to-end task)

III. Background

Belief state theory is undergirded by theoretical semantics research from the past half-century. Two key papers, Kiparsky and Kiparsky's *Fact* (1970) and Grice's *Logic and Conversation* (1975), appear as major inspirations for later linguistics research. Although our specific goal is much narrower in scope than what these papers discuss, they nevertheless act as guiding forces for our work and are worth describing here.

We begin with Kiparsky and Kiparsky. This paper postulates two syntactic categories for sentence predicates, *factive* and *non-factive*, and furthermore that the syntax of the predicate is predictable through whether or not we as readers can infer that the speaker presupposes their claim to be true. Let us start with an example:

Predicate Type	**Sentence**
Factive	It is odd that it is raining
Factive	I regret that it is raining.
Non-Factive	It is likely that it is raining.
Non-Factive	I suppose that it is raining.

In the factive examples above, it is clear from the form of the sentence that the speaker or author believes that it is raining. In the non-factive cases, we can make no such inference. In the factive examples, the author claims something about an event that is real according to that same author. In the non-factive examples, the author does not presuppose that the event is real. In the factive examples, the author expresses personal sentiments, i.e., surprise and regret, *about* the fact that it is raining, therefore qualifying the proposition that it is in fact raining as the type of presupposition falling under the factivity rule. A counter-example could be, "John's being ill is

impossible to imagine," because, even though John's illness is stated as objective truth, it is not presupposed by the author that John is in fact ill.

An excerpt of a list of factive and non-factive adjectives as shown in the paper appears below.

Factive	**Non-Factive**
Significant	Likely
Odd	Sure
Tragic	Possible
Exciting	True
Relevant	False
…	…

Currently, our research does not encompass the preservation of contextual information, including presuppositions, between sentences, as our corpora do not include annotations of contexts extending beyond a sentence at a time; however, we see deep learning detection of these sentence paradigms as an eventual goal toward which our current work will act as a stepping stone.

Grice also discusses how competent speakers make important inferences from what they hear based on identifiable patterns and structures, but, here, the patterns are more relevant to how one utterance follows the previous utterance, preserving relevance between them. By breaking the norms of conversational relevance, speakers can communicate ideas without explicitly stating them, thus introducing an *implicature*. Let us look at an example.

Example 1: A driver speaks to a local on the side of the road after running out of fuel.

A: I am out of petrol.

B: There is a garage 'round the corner.

If we assume that both A and B are following conversational norms, then we can deduce that B is not infringing on the common expectation that conversational contributions be relevant. This leads us to conclude that B thinks the garage around the corner is open and selling fuel, that A's lack of fuel is a problem that must be solved, and that A can solve this problem by going to the garage to buy fuel. These unspoken assumptions allow A to perceive B's response as relevant to what A had just said. These unspoken thoughts connecting A's speech to B's in a bond of relevance are the implicature.

Implicatures appear in discourse when, on the surface, one or more of a set of proposed guidelines for a conversation are not met for one of a number of reasons. This set of guidelines is called the Cooperative Principle, and its maxims are quantity, quality, relation, and manner. Grice presents some illustrative analogies to these maxims:

Maxim of Cooperative Principle	**Definition**	**Example**
Quantity	Be informative, but not overly so.	If I ask for four screws, I expect you to hand me four screws. Not three, not five.
Quality	Do not say things you know are false or lack evidence for.	If I am baking a cake and ask for sugar, I expect you to hand me sugar, not salt.
Relation	Be relevant.	If I am mixing the cake batter, I do not expect to be handed a fascinating book.

Manner	Be brief and unambiguous.	I expect a helper to be clear with what they are doing and to do so in a reasonable amount of time.

Grice's maxims, while precise, simply reflect the implicit ground rules governing how we as humans converse with one another. Essentially, if a speaker violates one of these maxims and there is no obvious justification, then an implicature has been introduced. One more example appears below for emphasis.

Example 2: Of a man known to have broken up all the furniture, one says "He was a little intoxicated."

Here, the speaker is flouting the quality maxim since he is presented with irrefutable evidence of the drunk man's high level of intoxication but nevertheless disregards the rule of not uttering low-quality information, or information known to be false (i.e., that he was not overly drunk). Therefore, a conversational implicature appears, being that the man was indeed overly drunk.

This research is important because it was the first to recognize that it is possible to precisely define surface features of speech that convey unspoken ideas, and, indeed, that is what this project aims to leverage with deep learning: predicting unspoken ideas from surface texts because these rules exist, despite their complexities.

IV. Language Understanding Corpus

Work began with the Language Understanding (LU) corpus. LU, appearing as a set of flat XML files, represents factuality annotations as spans: start and end nodes mark the beginning and ending tokens of an annotation (i.e., target), where the span may be between one and four tokens. Spans consisting of more than one token encapsulate compound nouns or noun phrases, typically referring to events. Each annotation has three possible labels, and is with respect to the author of the sentence; there are no nested sources in LU. See the below table for a summary of the three labels, noting that "factuality" here refers to the same concept as "factivity" in previous sections.

Label	Description
Committed Belief	The author is committed to the factuality of their utterance.
Non-Committed Belief	The author is not committed to the factuality of their utterance.
Not Applicable	No belief was expressed.

At first, LU's files were parsed into Python objects under a custom design the group called the Cognitive States Data Structure (CSDS). Each object represents a single annotation, with attributes such as the sentence containing the annotation, the annotation's target, and the label. This corpus is relatively simple, with all annotations being "author-only," meaning the source is always the author of the sentence itself rather than another source embedded in the sentence. John in "John said Mary is coming to dinner" would be an example of an embedded, or nested, source, which would be labeled as N/A since the reported belief is not the author's belief.

V. FactBank, Initial Experiment, and Source-Target Corpus Unification Database

The most difficult and fruitful work of this thesis centered around the FactBank corpus, a richer and more complex corpus than LU. Factbank diverges from LU in two key ways. First and most importantly, the data is stored relationally: while in flat files, the data are intended to be ported to a relational database and come with columns to be used as primary keys and foreign keys. We are using SQLite as the relational database engine. Second, Factbank supports embedded sources and preserves the nesting tree of embedded sources for each sentence. For the sentence "John said Mary is coming to dinner," the sources table would have two entries, AUTHOR to denote the author of the sentence (every sentence has one such entry) and one additional entry, John_AUTHOR, which denotes "John according to the sentence's author." Every training and testing example consists of a unique combination of a source identifier, a target identifier, and a belief value. FactBank's possible belief values appear below.

Belief Value	Description
CT+ (Certainly Positive)	According to the source, it is certainly the case that _____.
PR+ (Probably Positive)	… it is probably the case that _____.
PS+ (Possibly Positive)	… it is possibly the case that _____.
CT- (Certainly Negative)	… it is certainly not the case that _____.
PR- (Probably Negative)	… it is probably not the case that _____.
PS- (Possibly Negative)	… it is possibly not the case that _____.
Uu (Fully Underspecified)	The source does not know the factual status of the event or does not commit to it.

Previous work using FactBank has omitted the use of nested-source training examples due to the native database design being adversarial to the goal of using one SQL query to collect all of the necessary data. The below query illustrates the complicated series of joins necessary to collect the author-only annotations.

```
Unset
SELECT DISTINCT f.eText, f.relSourceText, s.file, s.sent,
t.tokLoc, f.factValue, o.offsetInit, o.offsetEnd, o.sentId
                    FROM sentences s
                    JOIN tokens_tml t
                        ON s.file = t.file
                               AND s.sentId = t.sentId
                    JOIN offsets o
                        on t.file = o.file
                            and t.sentId = o.sentId
                                and t.tokLoc = o.tokLoc
                    JOIN fb_factValue f
                        ON f.sentId = t.sentId
                               AND f.eId = t.tmlTagId
                               AND f.eText = t.text
                    WHERE f.relSourceText = "'AUTHOR'"
                    order by s.file, o.sentId;
```

Note that simply removing the "where" clause fails to yield a fully correct result set, and that additional tweaking of the above query only furthered us from the result set we wanted. Furthermore, the "according to" relation established by the previously-mentioned underscore notation for nested sources made little sense for our purposes; we desired a tree representation. Clearly, more work was needed to preprocess FactBank in order to fully utilize the nested-source data.

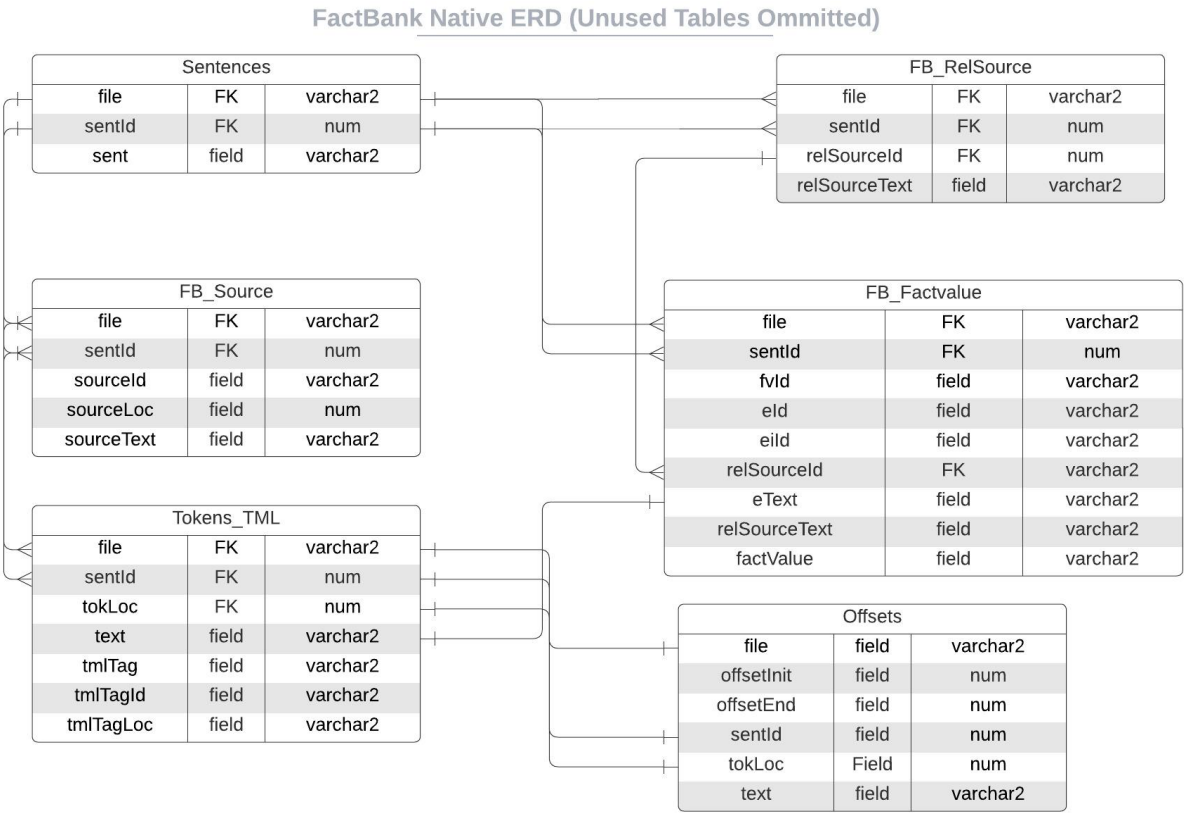
Before proceeding with this work, we ran an experiment using just FactBank's author-only annotations. The composition of FactBank is such that over ninety percent of the

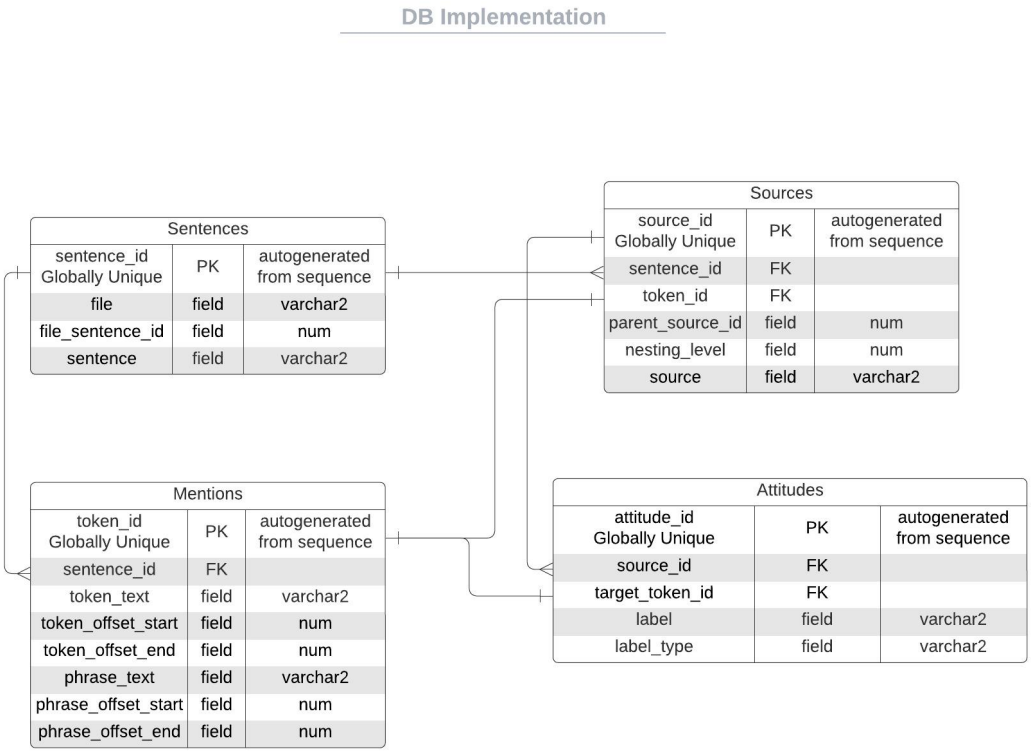
training examples are labeled CT+ or Uu: since there was not enough training data for FactBank's other classes within the author-only training examples, our experiment did not find results for them. It is also important to note that the CT+ examples were much more numerous than the Uu ones.

Overall, the results of this experiment were consistent with previous findings. We did not observe any significant improvement or decline. See the results section for detailed experimental results of fine-tuning a large language model to classify Factbank author-only annotations. More experimental results for FactBank appear in Murzaku et al.

With the Factbank author-only experiment complete, the next step was to bring in the nested-source training examples for additional fine-tuning, marking the second iteration of the codebase. As discussed, we already had a SQLite representation of FactBank, but the structure of the database, as presented by FactBank's authors, turned out to be extremely convoluted and poorly suited for our purposes.

This sparked an idea, not only to make using FactBank easier but to streamline future work: designing a master SQLite database schema into which we could then port each respective corpus we wish to analyze, while preserving data from original corpora as much as possible. This would afford a plethora of advantages, chiefly the standardization of SQL queries across corpora. Another advantage is an easier onboarding process when others join the project and want to understand our data quickly. Otherwise, one would need to learn each corpus' structure in detail, including its storage format and possible dependencies. An entity-relation diagram of the relevant tables in FactBank's native format, as well as our custom schema, appears below.


FactBank Native ERD (Unused Tables Ommitted)
Sentences
file FK varchar2
sentId FK num
sent field varchar2
FB_RelSource
file FK varchar2
sentId FK num
relSourceId FK num
relSourceText field varchar2
FB_Source
file FK varchar2
sentId FK num
sourceId field varchar2
sourceLoc field num
sourceText field varchar2
FB_Factvalue
file FK varchar2
sentId FK num
fvId field varchar2
eId field varchar2
eiId field varchar2
relSourceId FK varchar2
eText field varchar2
relSourceText field varchar2
factValue field varchar2
Tokens_TML
file FK varchar2
sentId FK num
tokLoc FK num
text field varchar2
tmlTag field varchar2
tmlTagId field varchar2
tmlTagLoc field varchar2
Offsets
file field varchar2
offsetInit field num
offsetEnd field num
sentId field num
tokLoc Field num
text field varchar2



It is clear that the new schema is easier to comprehend and use for us as well as the research community at large, considering the care that was taken to maintain abstraction in the new schema. Many source-target-factuality corpora will fit into this mold.

VI. Source-Target Corpus Unification Database: Loading in Factbank

Once we finalized the master schema, we endeavored to load in Factbank from the previous database while preserving the validity of each training example, including those with a nested source. Our first attempt at this was to tweak and reapply the query we had used for the author-only experiment, removing the query's author-only restriction and filling in the appropriate tables for each row in the result set. However, the nature of the key relationships in Factbank required much more meticulously designed and granular logic than what that query provided.

Rather than fill in the master schema directly from the result set drawn from the native database, we needed to select one sentence and then iterate over the relevant information contained within it, starting with the author annotation, since it is always present (every sentence has an author), and working through each additional level of source nesting, in order. On each subsequent nesting level, we would iterate over all available sources, making inserts to our sources table. Then, for each source, we would iterate over all of its associated targets, pulling out each target's character offsets and label before inserting them into our mentions and attitudes tables, respectively. This redesign of the logic to populate the database was critical in order to successfully pull out the pieces of each annotation (source, target, character offsets, label) from different parts of the native database, which the single-query solution could not handle due to its reliance on straight iteration over a surface-level result set. Importantly, this work allowed us to preserve relationships between parent and child sources as a feature of the database rather than relying on embedded text representations (i.e., John_AUTHOR).

Of course, a key feature of the unified database is that differing data representations between corpora are consolidated in the act of loading them into the database. As much as possible, the goal is to preserve native data, creating synthetic data where necessary to fill in discrepancies between corpora. For Factbank, this meant creating n-gram spans for source and target annotations, since other corpora, such as MPQA, contain them natively. This ended up being a complicated task given the theoretical work of defining a span in precise terms. Ultimately, we utilized the SpaCy library in Python to develop customized logic to traverse the parse tree for each head-to-span conversion, starting from the head token provided by Factbank and locating an ancestral token whose part of speech and left and right edges (leftmost and rightmost dependents) are such that a satisfactory span is emitted (see the code for details).

We define a span as a noun or verb phrase containing information about an event. The boundaries of a span are determined by the leftmost and rightmost dependents of the root word contained in the phrase. The only special case we handle is where a trailing conjunct is encountered on the right side of a span, which left untreated was making spans too wide, often to include the entire sentence containing the annotation. Fixing that required additional customized logic to exclude trailing conjuncts. See below for some examples.

Sentence	Token of Interest	Span
John said Mary is coming to dinner and that we are eating pizza.	coming	Mary is coming to dinner
John said Mary is coming to dinner and that we are eating pizza.	are	we are eating pizza

Separate from the issue of spans was the unification of Factbank's label scheme with MPQA's, which includes a reported belief class, ROB. A reported belief is a belief state that a source attributes to a nested source, for example in news reports: "The reporter reminded the

politician of the recent CNN exposé concerning his dubious campaign finance practices." Here, the reporter is conveying the belief held by CNN, which is that the politician is corrupt, but the reporter themselves does not express their own belief in the act of expressing CNN's belief. In this example, there are three sources who all share the same target: the author, the reporter, and CNN. The target is the belief that the politician is corrupt. For the author and the reporter, the label is ROB, while for CNN, it is Uu.

In Factbank's native form, these annotations are lumped together with Uu, so we had to design custom logic to separate them. If the nesting level of the source is greater than zero (i.e., not an author annotation), and the label is not Uu, we traverse the tree of parent sources from the current one all the way to the top, switching all Uu labels to ROB. This task ended up relabeling ~2700 examples to be part of the reported belief class.

VII. Source-Target Corpus Unification Database: Loading in LU

Porting the CSDS object representation for LU mentioned in section (IV) to the database was, fortunately, much simpler than the corresponding task for FactBank. LU's targets include both unigram heads and n-gram spans. However, LU's spans do not conform to our definition of a span as described in the previous section as they only encapsulate complex nouns and multi-token identifiers. Therefore, we converted all of LU's native spans to unigram heads, before filling in synthetic spans for all annotations. The inverse of head-to-span, span-to-head, therefore appeared as a necessary task for span-based annotations. SpaCy made this task very easy, though: extracting the head token from a span is a one-line operation in SpaCy. With no nested sources present in LU, the rest of the algorithm for converting the CSDS object representation to the database representation was very straightforward.

VIII. Methods

LU, once in our database, became the focus of our experimental work. All targets were reduced to unigrams due to early observations that using spans was negatively impacting results in a severe manner. We began by defining two tasks, drawing from insights presented by Zhang et al. and Murzaku et al.: first, an end-to-end generative task, in which the model receives the raw input sentence and outputs either the same sentence with predicted in-text annotations added by the model or a sequence of (target, label) tuples, and second, a typical classification task where each token in the input sentence appears as a separate training example, and the output is the relevant label for annotated tokens and O (other) for all other tokens. Let us define these and other terms.

An end-to-end task, in the context of machine learning, is a task in which the model provides a fully-functional, complete solution without any additional context needed. End-to-end tasks differ significantly from classification tasks since, in the end-to-end case, the model is attempting to perform a broader and therefore more difficult task in outputting a full solution instead of simply a label, as is done for classification.

In the following experiments, we used two models: Flan-T5 and BERT. Flan-T5 is a 220-million parameter text-to-text transformer model optimized for reasoning and question-answering tasks. BERT is a 110-million parameter transformer model optimized for masked language modeling (hiding a word in a sentence and then predicting it) and next sentence prediction (predicting whether or not one sentence follows another in a larger text). We chose Flan-T5 in order to match up with the model choice made by Murzaku et al. and Zhang et al.; we seek to validate some of their results. One important facet of Flan-T5 is that it performs best when provided with a task prefix such as "classify: " for a classification task. BERT was chosen

because it acts as a universal baseline for NLP and is especially skilled at classification tasks despite its smaller size.

For the end-to-end tasks, we borrowed two data formatting techniques, hereon referred to as paradigms, from Zhang et al. in order to validate their effectiveness on new data. Between the two paradigms, called extraction and annotation, the input is always the sentence itself. Furthermore, for both end-to-end tasks, all of the annotations for a given sentence are collapsed into a single training example, connecting to the general rule of end-to-end tasks being harder than classification tasks: the model has to make more predictions at once, with less information. The two paradigms differ, however, in their output formats. For extraction, the model outputs a series of (target, label) tuples, where the source is implied to always be the author since the experiments were all performed on the LU corpus. For annotation, the output is again the entire sentence, but with targets replaced with a regular expression of the form [target|label], corresponding to a literal annotation of the output by the model.

Turning to the classification task, we introduced the Other ('O') label in order to make the task somewhat comparable to the end-to-end task in that the model will not only have to predict which label to assign to a target, but whether or not that target should be annotated at all. In this sense, the 'O' label represents any token in LU that was not annotated in the original corpus, forming a supermajority class.

See the table below for a summary of these tasks with examples.

Task	Model(s)	Paradigm	Example Input	Example Output
End-to-End	Flan-T5	Extraction	John said Mary is coming to dinner.	(coming, CB)
End-to-End	Flan-T5	Annotation	John said Mary is coming to dinner.	John said Mary is [coming|CB] to dinner.
Classification	Flan-T5, BERT	N/A	… Ex 1. coming|||John said Mary is coming to dinner. Ex 2. to|||John said Mary is coming to dinner. …	… Ex 1. CB Ex 2. O …

With these three tasks defined, we upgraded the codebase presented by Zhang et al. and modified by Murzaku et al. to support the LU corpus and the classification task, both of which did not work natively. Furthermore, we implemented five-fold cross-validation for all experiments, so that, between all of the folds, the model would train on every single example, thus providing a well-balanced result. Crucially, file boundaries were respected between folds such that a file's annotations were never split between two folds. This prevented the model from getting distracted by learning features of the files themselves.

For the two end-to-end tasks, we used the same normalization logic to extract predictions from generative outputs as Zhang et al. The best prefix for these tasks ended up being "lu factuality: ". Additional speculation regarding model prefixes on this task is not useful since previous work (Murzaku et al.) investigated this extensively and did not find a significantly better-performing prefix.

Regarding the classification task, some slight database modifications were required to introduce the 'Other' (O) label. Most of the work here had to do with adjusting offset boundaries for those annotations in LU where the native target is an n-gram span versus a unigram head, since the SpaCy tokenizer, which handled the span-to-head task, differs slightly from the tokenization implied in the way LU presents offset boundaries for spans. With that fixed, generating the data in the correct format was easy. The prefix for this task was simply "classify: " (only applied to Flan-T5 - no prefix was used with BERT).

All three tasks were run through Flan-T5, whereas the classification task was also run through BERT. Targets for all experiments were represented as unigram heads, and all experiments were run three times, with each run assigned one of the following random seeds, which is provided to the model in order to ensure that changes to results between runs will be entirely caused by changes in the data: 7, 21, and 42. Cross-validation was implemented within each of these three runs. See part (B) of the experimental results section for results.

IX. Experimental Results

A. Factbank: Author-Only Initial Results (Pre-Database)

Classifier	Metric	Value
CT+	Precision	0.932
CT+	Recall	0.957
CT+	F1 Score	0.945
Uu	Precision	1.0
Uu	Recall	0.158
Uu	F1 Score	0.273

To interpret these data, we must first understand how these metrics work and what they tell us about BERT's performance. For a given classifier, a true positive arises when the predicted label is correct, and a false positive when it is not. Furthermore, a true negative occurs when the predicted label matches the true label but it is not the one we care about, and, finally, a false negative appears when the true label is relevant but the predicted one is incorrect. Precision is defined as the number of true positives divided by the total number of true positives and false positives, representing the percentage of the time that BERT correctly labeled training examples within all of its predictions for a given label. On the other hand, recall is calculated as the number of true positives divided by the total number of true positives and false negatives, meaning the percentage of the time BERT actually caught training examples with a given label in the sample. Finally, the F1 score is the harmonic mean of precision and recall, acting as a combinatory metric of the two.

Somewhat unsurprisingly, the performance of the CT+ class was high, since this was the majority class (i.e., the majority of training examples had CT+ as the true label), thus giving deep learning more data to work with. The Uu class performed worse, although the precision score of 1.0 was good; this can be interpreted as, for every training example, if it was predicted to be Uu, it was correct. The low recall score makes sense due to the Uu class being small: if the model predicted Uu, it was likely to be correct, but the model also missed many examples of Uu, giving it a low recall score. FactBank's high number of classes combined with the small size of the Uu class coalesce to produce this score (many more false negatives than false positives).

B. LU: All Results

Note: For the following four tables, macro-f1 refers to the averaging of results of machine learning prediction across three different seeds for the initialization of the random number generator. The results for each seed are already averaged across five folds each.

B.I. End-to-End Extraction Paradigm on Flan-T5 (Normalized)

Classifier	Metric	Value
Committed Belief	Macro-F1	0.706
Average of All Labels	Macro-F1	0.405

B.II. End-to-End Annotation Paradigm on Flan-T5 (Normalized)

Class	Metric	Value
Committed Belief	Macro-F1	0.730
Average of all labels	Macro-F1	0.441

Between the two tables above, it is clear that the annotation paradigm significantly outperforms the extraction paradigm; this is consistent with the findings of Zhang et al. and Murzaku et al. As expected, minority classes, whose scores are omitted here due to their being very low, performed worse due primarily to the very small number of available training examples in the corpus.

B.III. Classification with Flan-T5 (Raw)

Classifier	Metric	Value
Committed Belief	Macro-F1	0.746

Non-Committed Belief	Macro-F1	0.397
Non-Applicable	Macro-F1	0.611
Other	Macro-F1	0.967
Average of all labels except other	Macro-F1	0.585

B.IV. Classification with BERT (Raw)

Classifier	**Metric**	**Value**
Committed Belief	Macro-F1	0.742
Non-Committed Belief	Macro-F1	0.477
Non-Applicable	Macro-F1	0.6
Other	Macro-F1	0.967
Average of all labels except other	Macro-F1	0.606

On the classification task, we observed slightly higher but overall very similar performance compared to the end-to-end tasks; between Flan-T5 and BERT, the majority class performance was identical. Interestingly, minority-class performance was far better here than on the end-to-end tasks, yielding a much-improved macro average F1 score. Overall, BERT outperformed Flan-T5 on the classification task.

It is important to note that LU remains largely unexplored. This means that these results are difficult to compare to previous results, although they can be compared to the FactBank experiments using the same methods as we did.

X. Conclusions and Insights

A. FactBank: Author-Only Initial Experiment (Pre-Database)

The data presented here do not lead to conclusions as impressive as those in Murzaku et al. since the experiment in question was re-run with methods more closely resembling those followed for the LU experiments, namely drawing data from the unified database instead of the native format. Nevertheless, useful some conclusions can be drawn.

From the observed metrics for the majority class, CT+, we can conclude that BERT is indeed able to correctly infer the belief state of the author when that belief state is CT+. The comparatively poor performance of the Uu class may be due to the fact that the Uu class is simply not large enough to match the CT+ class, but we suspect there is more behind Uu's performance. In FactBank, the Uu class acts as a catch-all for annotations that generally do not express a belief; the linguistic features of these annotations may vary widely, however. This is in contrast to the CT+ class, where the syntactic and grammatical structures of annotations are shared; this makes learning easier. To improve performance, the Uu class should be reexamined to see whether some subset other than the previously-discussed reported belief class could be relabeled.

B. LU: All Results

B.I and B.II: End-to-End Tasks on Flan-T5

The most salient conclusion to be drawn from this set of experiments is that Flan-T5 does better with tasks that ask for generative outputs, which was done here with the annotation paradigm: the correct output contains the entire sentence. It is important to note that all classes saw improved performance, not just the majority class.

We also observed that Flan-T5 does not do well with span-based annotations or a mix of unigrams and n-grams. The best performance by far occurred when all annotations used unigrams. This suggests that including non-essential tokens in annotations fed to the model only serves to confuse it and lower accuracy. Further research on the effectiveness of spans in belief state inference tasks is needed.

B.III and B.IV: Classification Task on Flan-T5 and BERT

Comparing the results for classification to the results for end-to-end, it is clear that Flan-T5 performs much better on classification. This makes sense from the perspective of task difficulty: in the end-to-end tasks, the model only gets one attempt to correctly predict all of the labels for annotations contained in a sentence and where they appear in the sentence. This is in clear contrast to the classification task, where a small phrase, or, in our case, a unigram head, is provided alongside the full input sentence. The model only needs to choose a label. Even if the prediction is wrong, that mistake will not affect other predictions, since they are in separate examples. The same cannot be said for end-to-end predictions, since all predictions for a given sentence are contained in a single output.

The almost-perfect score for the 'O' label is both expected and insignificant. Per the methods, any token not contained in an annotation is assigned this label, therefore making it a supermajority class.

XI. Future Work

A. Plans to Improve Experimental Results

The main deficiency in our results was in regard to minority class performance. We have two ideas for improvement.

The first idea is to use a few-shot approach. Since the set cardinalities of LU's minority classes are so low, it would make sense to form support sets from them and then run all minority-class predictions through a few-shot system, while either running majority-class predictions through the same system (requiring another support set for the majority class) or a different LLM altogether.

Another idea is to create synthetic minority-class data. This would entail manipulating existing majority-class annotations by substituting words that indicate a committed belief with words that indicate a non-committed belief, for example. WordNet appears as a useful tool for the word replacement task since it houses vast semantic relation data.

B. Additional Lines of Inquiry

While in the process of developing the logic for the head-to-span task, it occurred to us that there is more to explore in the realm of parse trees. Although dependency parsing is a thoroughly well-studied area and state-of-the-art accuracy hovers around 97 percent (i.e., the best dependency parsers today produce an incorrect parse approximately 3% of the time), we wonder whether an LLM trained on a set of head-to-span data could do a better job identifying spans from heads (input head, predict span), and conversely, heads from spans (input span, predict head), than other dependency parsing tools such as SpaCy. Note that the proposed idea is a sub-task of the general dependency parse task, so we are not seeking to improve on the 97% accuracy of parsing overall, but rather a part of that task.

Looking at the majority classes between FactBank and LU, it is clear that the annotations in these classes share syntactic and grammatical features that the annotations in classes like Uu do not. With that in mind, we are curious as to whether we could identify additional sub-classes within Uu that consistently share enough syntactic and grammatical features such that we could separate annotations holding these features into additional classes, akin to what we did with the reported belief class.

XII. References

Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., … Wei, J. (2022). Scaling Instruction-Finetuned Language Models. doi:10.48550/ARXIV.2210.11416

Diab, M., Dorr, B., & Levin, L. (2009). *Language Understanding Annotation Corpus*. Linguistic Data Consortium. https://catalog.ldc.upenn.edu/LDC2009T10

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

John Murzaku, Peter Zeng, Magdalena Markowska, and Owen Rambow. 2022. Re-Examining FactBank: Predicting the Author's Presentation of Factuality. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 786–796, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Sauri, R., & Pustejovsky, J. (2009). *FactBank 1.0*. Linguistic Data Consortium. https://doi.org/10.35111/tk5m-zw90

Wenxuan Zhang, Xin Li, Yang Deng, Lidong Bing, and Wai Lam. 2021. Towards Generative Aspect-Based Sentiment Analysis. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on*

Natural Language Processing (Volume 2: Short Papers), pages 504–510, Online. Association for Computational Linguistics.