StFX-NLP at SemEval-2024 *Task* 9: BRAINTEASER: Three Unsupervised Riddle-Solvers

Ethan Heavey, James Hughes, Milton King
St. Francis Xavier University
{eheavey, jhughes, mking}@stfx.ca

Abstract

In this paper, we explore three unsupervised learning models that we applied to *Task* 9: BRAINTEASER of SemEval 2024. Two of these models incorporate word sense disambiguation and part-of-speech tagging, specifically leveraging SensEmBERT and the Stanford log-linear part-of-speech tagger. Our third model relies on a more traditional language modelling approach. The best performing model, a bag-of-words model leveraging word sense disambiguation and part-of-speech tagging, secured the 10^{th} spot out of 11 places on both the sentence puzzle and word puzzle subtasks.

1 Introduction

Riddles often exploit the commonsense of the solver to lead them astray, subverting expectations with it's answer. For example, the riddle "A young girl fell off of a 20 foot ladder but wasn't hurt. How? *She fell off of the bottom rung.*" leads the solver astray by including the height of the ladder in the initial question, tricking one into latching onto misleading information. *Task* 9: BRAINTEASER (Jiang et al., 2024) presents riddles to a predictive model and asks the model to choose one of four answers to the riddle, in the hopes of bridging the gap between vertical and lateral thinking (Waks, 1997) within language models. The data provided for the *Task* is written in English and was obtained from public websites by utilizing web crawlers (Jiang et al., 2023).

The three models we employ to solve this task all apply an unsupervised learning approach, with two of the three models leveraging word senses and part-of-speech tagging to aid in their predictive capabilities. We wanted to leverage the senses of the nouns in the question and in each possible answer as we hypothesized that the senses present in the question and each answer may aid our models in piercing the proverbial commonsense veil that makes brainteasers and riddles difficult to begin with.

Our best approach, the bag-of-words model, landed us in 10^{th} place out of 11 places in the "overall" results of both subtasks. While 13 teams competed, two teams tied for both 2^{nd} and 4^{th} place in the sentence subtask, two teams also tied for both 1^{st} and 11^{th} place in the word subtask results.

Our code can be found on Github[1].

2 Background

BRAINTEASER places emphasis on the ability of a predictive model to use vertical and lateral thinking. Vertical thinking leverages logic and rationality to perform a sequential analysis of a problem, whereas lateral thinking (or "thinking outside the box") leverages creativity to solve problems. The *Task* is divided into two subtasks — sentence puzzles and word puzzles. We applied our models to both, with each subtask requiring vertical and lateral thinking to solve. Figure 1 breaks down how sentence and word puzzles can be solved with lateral thinking. The train of thought labeled with a red "X" demonstrates logical thinking based on the information available at the time, whereas the alternate thought process — the line of thinking that allows the solution to be derived — displays how lateral thinking can affect the answer to a riddle as more context is provided.

The dataset associated with the *Task* presents each sample as a question and four possible answers. Table 1 shows an example of both a sentence puzzle question and its possible answers, and a word puzzle question and its possible answers. Each sample also has two variants; a semantic reconstruction and a context reconstruction. These reconstructions are designed to further test a model's reasoning ability.

[1] https://github.com/VeiledTee/BrainTeaser

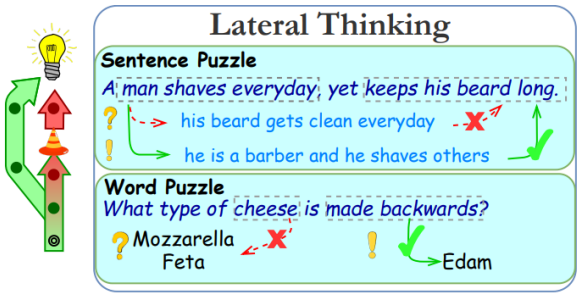

Figure 1: An example of how lateral thinking can be used to solve sentence and word puzzles. Figure taken from BRAINTEASER system paper (Jiang et al., 2023).

Whilst the training and development sets contain extra information regarding the correct answer, our unsupervised approaches only required the test set. Not using the labeled training and validation data, while limiting our models, allows them to be more versatile in situations where labeled data is not available.

Word Sense Disambiguation (WSD) is a natural language processing (NLP) task that involves determining the correct meaning or sense of a word within a given context (Navigli, 2009). Many words in natural language have multiple senses, and WSD aims to identify the intended sense of a word in a specific sentence or context. This is used in various language processing applications, such as machine translation, information retrieval, and text summarization. We employ WSD by leveraging SensEmBERT (Scarlini et al., 2020), coupled with WordNet (Fellbaum, 1998) to disambiguate the sense of a token in a particular context.

SensEmBERT is a knowledge-based approach to WSD that produces high-quality sense embeddings. WordNet is a large lexical database that organizes words and their meanings into sets of interlinked synonyms called synsets.

We leverage part-of-speech (POS) tagging in order to determine which tokens in each question and answer are nouns we can determine the sense of. We employ the English version of the Stanford Log-Linear POS Tagger[2] (Toutanova et al., 2003) — which leverages dependency networks to aid in tagging tokens — in this work. For the purposes of our work, we only work with nouns — tokens whose tag begins with "NN".

[2] https://nlp.stanford.edu/software/tagger.shtml

3 System Overview

The following is a description of each approach we took in an attempt to solve the *Task*. We implemented a bag-of-words, language modelling, and a sense comparison approach. The language model at the core of all three of our approaches is `bert-large-cased` (Devlin et al., 2018), the same model leveraged by Scarlini et al. (2020) in the creation of SensEmBERT.

3.1 Bag of Words with WSD Approach

Our bag-of-words (BOW-WSD) model combines POS tagging with WSD to create a bag of words for the question and each possible answer. When presented with a question (q), the model creates a list containing the most prevalent sense for each noun in the question — q_senses — by leveraging Algorithm 1. Note; in this algorithm, it is necessary to concatenate the embedding of each noun to itself in order to match the format of the WordNet senses, allowing said WordNet senses to be compared to and leveraged. From q_senses, we create q_bag by removing all stop and duplicate words. Token order and context is preserved during the generation of q_senses but not for the creation of q_bag.

The process used to create q_bag is then repeated four times — once for each possible answer — creating five bags of words in total, one q_bag and an $answer\_bag$ for each of the four answers. Each $answer\_bag$ is compared to q_bag through an overlap calculation — the number of common tokens across both bags — shown in Equation 1. For example, if q_bag is "[hair, shave, beard, cut, trade]" and one of the $answer\_bags$ is "[trade, cut, hair, someone]", the overlap score would be 0.667 — three overlapping tokens of nine possible tokens. The $answer\_bag$ with the highest overlap score is predicted to be the correct answer.

$$\text{avg\_overlap} = \frac{2 \cdot (|\text{bag1} \cap \text{bag2}|)}{(|\text{bag1}| + |\text{bag2}|)} \quad (1)$$

3.2 Language Modelling Approach

In the example shown in Table 1, the correct answer can be read as a natural continuation of the question — contrary to the other possible answers which do not make logical sense if appended onto the end of the question. We explore this intuition with our language modelling approach, which takes each answer, concatenates it to the end of the question, and calculate the probability of the text from

Question	Choices
Sentence Puzzle Example	
A man shaves everyday, yet keeps his beard long.	*He is a barber.* He wants to maintain his appearance. He wants his girlfriend to buy him a razor. None of the above.
Word Puzzle Example	
What part of London is in France?	*The letter N.* The letter O. The letter L. None of the above.

Table 1: An example of a sentence puzzle and a word puzzle from the BRAINTEASER dataset. The correct answer for each puzzle is in italics.

Algorithm 1: WordNet Sense Extraction

Input: Input sentence
Output: WordNet senses of nouns in the sentence
`bert-large-cased` tokenizes input
Perform POS tagging on tokenized input
$filtered\_nouns \leftarrow$ nouns from the POS tagging results
$final\_senses \leftarrow []$
for n *in* $filtered\_nouns$ **do**
    Concatenate the noun's token embedding to itself /* This format matches that of WordNet, permitting querying */
    Search WordNet for the most similar sense key using cosine similarity
    Use sense key to retrieve WordNet sense of n
    Append n_sense to $final\_senses$
Return $final\_senses$

each answer following the question using BERT (`bert-large-cased`)[3]. The predicted answer is the one associated with the largest probability.

3.3 Sense Comparison Approach

In this approach we leverage an unsupervised WSD model that makes predictions by comparing the senses of nouns. Once the primary sense of each noun in the question is identified, we utilize the `bert-large-cased` model to retrieve the embedding of the `[CLS]` token for each identified sense. This procedure is replicated for every potential answer, and the cosine similarity is employed to compute a similarity score for each pairing of `[CLS]` tokens between the senses of the question and those of each individual answer. Subsequently, these similarity scores are aggregated and averaged based on the number of senses being assessed in the current computations, both for the question and the answer. The predicted answer is the one with the highest average similarity score. Algorithm 2 outlines the steps this approach takes in further detail.

Beyond the data provided by the *Task* organizers, we leveraged the English stop words available through the NLTK Python library[4] (Bird et al., 2009), and the senses provided by WordNet[5] (Fellbaum, 1998).

4 Experimental Setup

As previously mentioned, we only use the test set in our experiments. Due to the unsupervised nature of

[3] https://huggingface.co/bert-large-cased

[4] https://www.nltk.org/

[5] https://wordnet.princeton.edu/

Algorithm 2: Sense Comparison

Input: question, list of four possible answers
Output: Predicted answer
$q\_senses \leftarrow$ WORDNETSENSEEXTRACTION(*question*)
$q\_CLS \leftarrow$ [embedding for $sense$ in q_senses] `// calculated by bert-large-cased`
$answers \leftarrow [choice_1, choice_2, choice_3, choice_4]$
$answer\_similarity \leftarrow []$
for a *in* $answers$ **do**
    $a\_senses \leftarrow$ WORDNETSENSEEXTRACTION(a)
    $a\_CLS \leftarrow$ [embedding for $sense$ in a_senses] `// calculated by bert-large-cased`
    $total\_similarity \leftarrow 0$;
    for $q\_CLS\_embedding$ *in* q_CLS **do**
        for $a\_CLS\_embedding$ *in* a_CLS **do**
            $similarity\_score \leftarrow$ COS_SIM$(q_CLS_embedding, a_CLS_embedding)$;
            $total\_similarity \leftarrow total\_similarity + similarity\_score$;
    $answer\_similarity[i] \leftarrow \frac{total\_similarity}{len(q\_CLS) \cdot len(a\_CLS)}$
$max\_index \leftarrow$ index of max element in answer_similarity
Return answers[max_index]

our approaches, the labels are not required to train our models as none of them had hyperparameters to tune.

4.1 Libraries used

Table 3 shows the Python libraries and their versions used for this *Task*. Python version 3.10.11 was used. The full `requirements.txt` file is available in our GitHub repository[6] for the project.

4.2 Evaluation Measures

The *Task* uses six metrics for both the sentence and word puzzles — 12 total — of metrics to evaluate a model's ability to solve brainteasers. The three different types of questions (original, semantic reconstruction, context reconstruction) were evaluated individually and in two groups. For a model to predict a sample in one of the groups (original and semantic reconstruction, original and semantic reconstruction and context reconstruction) correctly, all of the samples in said group must be predicted correctly.

5 Results

The performances of our models, the provided baseline models, and the best performing models submitted to this *Task* are found in Table 2.

Our BOW-WSD model (Section 3.1), the best performing of our three approaches, was able to surpass the RoBERTa-L baseline in 2 of 6 of the sentence puzzle categories, and outperforms the same baseline on 5 of 6 of the word puzzle categories. BOW-WSD outperforms or comes very close to outperforming the RoBERTa-L baseline in both "Overall" categories. The performance of our unsupervised models didn't approach the ChatGPT or Human baselines in any category. The closest our models got to the ChatGPT baseline was in the original word puzzle category with a difference of 0.155, whereas the closest our models got to the Human baseline was in the context sentence puzzle category with a difference of 0.469. The numbers achieved by our BOW-WSD model netted us 10^{th} place overall in the sentence puzzle subtask.

We suspect the relationship between the tokens in the question senses and the tokens in the correct answer's senses allowed our BOW-WSD model to outperform our other approaches. Using the sentence puzzle in Table 1 as an example, the WordNet sense of the noun "barber" (available below) from the correct answer has two tokens that overlap with the question, leading to this answer achieving a higher score than other nouns that don't overlap.

> a hairdresser who cuts hair and shaves beards as a trade

Our language modelling approach outperformed the RoBERTa-L baseline in 4 of 6 of the word puzzle categories, but did not perform well in any of

[6] https://github.com/VeiledTee/BrainTeaser

	Sentence Puzzle						Word Puzzle					
Test set	**Original**	**Semantic**	**Context**	**Orig. + Sem.**	**Orig. + Sem. + Con.**	**Overall**	**Original**	**Semantic**	**Context**	**Orig. + Sem.**	**Orig. + Sem. + Con.**	**Overall**
Best overall	1.00	.975	.925	.975	.900	.967	.969	.938	1.00	.938	.938	.969
Human	.907	.907	944	.907	.889	.920	.917	.917	.917	.917	.900	.917
ChatGPT	.608	.593	.679	.507	.397	.627	.561	.524	.518	.439	.292	.535
RoBERTa-L	.435	.402	.464	.330	.201	.434	.195	.195	.232	.146	.061	.207
BoW	.425	.400	.475	.350	.200	.433	.406	.219	.344	.125	.063	.323
LM	.225	.200	.375	.075	.050	.267	.438	.250	.500	.125	.031	.396
SC	.175	.200	.350	.175	.125	.242	.156	.063	.219	.063	.031	.146

Table 2: The accuracy scores achieved by our models (Bag-of-Words, Language Model, and Sense Comparison) on each sub-category of the test dataset. Approaches in gray are shown for comparison: the best scoring participant model for each individual category; the participant model that performed best in both the sentence and word puzzle subtasks; and the organizer's ChatGPT, RoBERTa-L, and Human baselines.

the sentence puzzle categories. We suspect that the way the word puzzles are structured lends more to the language modelling approach than the sentence puzzle structure as all the word puzzles in the test set are structured as questions — adding each answer to the end of the question can provide the language modelling approach with enough context to choose the correct answer. We believe the more succinct nature of the word puzzle problems allowed our language modelling technique to outperform our BOW-WSD model on 4 of 6 word puzzle categories, netting us 10^{th} place in the word puzzle subtask too.

Our sense comparison model unfortunately performed worse than all our models and the *Task* organizers' baselines. Our idea to leverage the senses of nouns in the sentences did not perform well when applied to this *Task*.

6 Conclusion

Whilst the best of our unsupervised models surpassed only one of the established baselines, we have been able to show that word sense disambiguation may have a place in riddle-solving models. Our BOW-WSD model performed better on the sentence puzzles, but our language modelling approach performed better on the word puzzle subtask. The inherent logical reasoning large language models obtain through the copious amount of training data they're trained on can be led astray by the information provided by a riddle. Leveraging word sense disambiguation we attempt to isolate the meaning of each noun and compare and contrast said meanings to those present in each possible answer.

In the future, we will explore other means of incorporating WSD models within our riddle-answering model along with an ensemble method. While our unsupervised approaches didn't perform well compared to other submitted models on the *Task* leaderboard, the senses of the nouns in each question and answer held information valuable enough to allow our models to surpass one of the three proposed baselines. Regarding our bag-of-words model, we will add a metric that penalizes an answer if the senses it displays are wildly different to those of the initial question. This penalty could reduce the impact red herrings typically found in riddles have on the BOW-WSD model's predictive abilities.

References

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of

deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Christiane Fellbaum. 1998. Wordnet: An electronic lexical database. In *International Conference on Lexical Resources and Evaluation*. LREC.

Yifan Jiang, Filip Ilievski, and Kaixin Ma. 2023. Brainteaser: Lateral thinking puzzles for large language model. *arXiv preprint arXiv:2310.05057*.

Yifan Jiang, Filip Ilievski, and Kaixin Ma. 2024. Semeval-2024 task 9: Brainteaser: A novel task defying common sense. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*, pages 1996–2010, Mexico City, Mexico. Association for Computational Linguistics.

Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM computing surveys (CSUR)*, 41(2):1–69.

Bianca Scarlini, Tommaso Pasini, and Roberto Navigli. 2020. Sensembert: Context-enhanced sense embeddings for multilingual word sense disambiguation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8758–8765.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 252–259.

Shlomo Waks. 1997. Lateral thinking and technology education. *Journal of Science Education and Technology*, 6:245–255.

A Appendix

Library	Version
`NumPy`	1.26.1[7]
`NLTK`	3.8.1[8]
`Transformers`	4.35.0[9]
`Scikit-Learn`	1.4.0[10]
`PyTorch`	2.1.0+cu118[11]

Table 3: Table of major Python libraries (and their versions) employed while working to solve the *Task*.

[7] https://numpy.org/doc/

[8] https://www.nltk.org/

[9] https://huggingface.co/docs/transformers/en/index

[10] https://scikit-learn.org/stable/

[11] https://pytorch.org/docs/stable/index.html