Span Labeling with Large Language Models: Shell vs. Meat

Phoebe Mulcaire
Duolingo
phoebe@duolingo.com

Nitin Madnani
Duolingo
nitin@duolingo.com

Abstract

We present a method for labeling spans of text with large language models (LLMs) and apply it to the task of identifying *shell language*, language which plays a structural or connective role without constituting the main content of a text. We compare several recent LLMs by evaluating their "annotations" against a small human-curated test set, and train a smaller supervised model on thousands of LLM-annotated examples. The described method enables workflows that can learn complex or nuanced linguistic phenomena without tedious, large-scale hand-annotations of training data or specialized feature engineering.

1 Introduction

Madnani et al. (2012) show that writers or speakers engaging in argumentative discourse do not simply enumerate their claims and evidence, but rather structure them in some manner for their argument to be convincing. Such discourse, therefore, might contain not only language expressing the core claims and evidence (the "meat" of the argument), but also language used to organize or support them (the 'shell"). The authors also propose approaches to automatically detect shell language in real-world examples of argumentative discourse, such as test-taker responses and political debates.

To illustrate the difference between "meat" and "shell", we provide a hypothetical test-taker response below discussing whether people learn better by being told what to do or shown what to do. Spans representing shell language are shown in bold while the core content of the argument is shown as plain text.

> **This is a very interesting topic for a debate. I would advocate the argument that** being shown what to do is the better option because people are visual learners. They learn better by watching than by just being listening to what someone else tells them. **While this may not apply to everyone, I think that** it certainly applies to the average joe. **For this reason, it is therefore clear that** being shown what to do is better.

In this paper, we build on the work of Madnani et al. (2012) by focusing more deeply on how shell language is used in responses written by test-takers for the Duolingo English Test (DET), a high-stakes English language proficiency test. Our goal is to build a finer-grained, accurate, and scalable shell detection pipeline for this use case, leveraging modern transformer-based approaches to power each stage.

We first detail our motivations for applying shell detection to test-taker responses (§2). Next, we discuss our annotation rubric for identifying shell language (§3) and our small-scale use of human annotation to validate and refine this rubric. We experiment with automatic annotation of test-taker responses using both non-reasoning & reasoning foundation models (§4), resulting in strong machine-human agreement rates. Then, we attempt to distill our annotations into a BERT model that would be cheaper & faster to deploy for operational use (§5), and provide additional discussion of our approach and error analysis (§6). Finally, we conclude with a comparison to related work (§7) and possible directions for future work (§8).

2 Motivation

English language assessments typically contain prompts asking test-takers to write open-ended responses as a demonstration of their writing proficiency. These prompts generally require arguing for/against a position with appropriate supporting

Proceedings of the 20th Workshop on Innovative Use of NLP for Building Educational Applications, pages 850–859

evidence, or relating a past event matching a high level-description (e.g. "talk about a time when..."). Given the nature of the writing tasks, these responses are likely to contain some amount of shell language, as illustrated by the sample response in the previous section.

A certain amount of shell is useful – necessary, in fact – to scaffold one's arguments and produce a comprehensible and convincing argument. However, we have observed that many test-takers overuse such language to artificially inflate response length and vocabulary sophistication – both of which can impact the accuracy of automated essay scoring systems.

In this paper, we want to reliably identify (and categorize) spans of shell language in test-taker responses, independently of whether it is used appropriately to connect and organize the text or misused to pad it out with formulaic phrases. Some possible applications of reliable shell detection would include:

- Detecting the use of memorized response templates and other bad-faith patterns that rely on shell language overuse,
- Developing an independent measure of content development (the "meat"), and
- Gaining insights into stylistic variance that may arise even in the absence of shell overuse

Achieving these goals would allow us not only to improve the robustness of automated assessment to a common strategy employed by test-takers to fool automated scoring systems but also to improve measurement of content and coherence.

3 Annotation Rubric

The starting point of our pipeline, and the foundation of our approach, is an annotation rubric which defines & categorizes shell text. We use this rubric for manual annotation as well as to bootstrap automatic annotations using large language models. Since Madnani et al. (2012) do not share any annotation guidelines, we construct our own rubric for identifying shell language in test-taker responses.

We relied on multiple rounds of human annotation to start with an initial draft of our shell annotation rubric[1] and refine it into its final form. Specifically, the authors first collaboratively annotated 11 test-taker responses using the draft rubric and made major revisions based on the ensuing discussions. Next, the authors independently annotated 50 additional responses based on the revised rubric to determine any remaining discrepancies which were then resolved in a curation session. No changes were made to the rubric after this point. Annotation, review and curation were performed using INCEpTION (Klie et al., 2018).

Given that our goal is to enable finer-grained analyses of shell language, our final rubric defines multiple shell categories, as described in the subsections below.

3.1 Category A: Discourse Markers/Linking Expressions

The shell language spans in this category are defined to be words and phrases that are either serving an organizational or discursive purpose. For example, ones that link sentences or paragraphs with the goal of progressing between ideas. However, single-word coordinating conjunctions like "because", "but", "and", etc. within sentences are not annotated as shell language. Examples of category A spans observed in test-taker responses include but are not limited to:

- `To begin with ...`
- `In conclusion ...`
- `Firstly ...`
- `Secondly ...`
- `In addition ...`
- `There are three examples of ...`
- `For example, ...`
- `That is because ...`
- `Expanding on the previous discussion ...`
- `This is another reason`
- `... in addition to the previous discussion`

As the examples show, this category is mainly defined by short phrases and expressions, not entire sentences. A complete sentence of shell-like material is more likely to be category B, which we describe next.

3.2 Category B: General/Vague Statements

This category consists of phrases or statements that are formal and/or impersonal in nature and add emphasis, reflection, or consideration of the prompt or topic under consideration but *without* any real

[1] To create the initial draft rubric, we employed few-shot prompting, supplying ChatGPT with a general description of shell language from (Madnani et al., 2012) along with 100 actual test-taker responses containing a range of shell language spans.

content. Spans of this extremely productive category are often employed as padding in bad-faith responses. A very small subset of observed examples is shown below.

- `It is imperative to recognize that ...`
- `... would be very significant for us`
- `In today's age ...`
- `Today in society, there is a heated on-going discussion on the topic of ...`
- `If you ask me I would say that the statement has both pros and cons.`
- `In this burgeoning epoch of science and technology, we are dwelling in the 21st century.`
- `There is a widespread worry that this will lead to a myriad of concern in the world.`

3.3 Category C: Prompt/Topic Restatement

This category contains sentences or chunks that simply restate the prompt or initial argument without any further development. We have observed that when a large part of the prompt is restated, the surrounding phrases are often from categories A or B. A few real-world examples are shown below with the corresponding prompt in parentheses. Note that only the category C spans are shown in bold; spans of any other categories are not shown.

- `Today in the society,there is a heated on-going on discussion on the topic that` **`due to the invention of cell phones, people can communicate via text messages`**`.`
  (*Due to the invention of cell phones, people can communicate via text messages. Describe the ways texting has changed how we communicate.*)

- `One of the most important trends in today's world is the sudden upsurge in the statement that` **`Acquiring new knowledge and skills doesn't always happen quickly`**`.`
  (*Acquiring new knowledge and skills doesn't always happen quickly. Do you think that patience is key when it comes to learning, or do you think it is possible to learn things quickly if you are motivated? Support your opinion with your personal experience and observation.*)

It must also be noted that not *all* mentions of the prompt should automatically be marked as shell. Specifically, we do not mark such a mention as shell if the response:

1. sufficiently restructures or paraphrases it (especially to use it as a topic claim) instead of just quoting or restating it, or

2. simply refers to to entities or noun phrases from the prompt in context.

As an example consider the span `... being focused on a single thing is more likely to lead to higher productivity` in a response to the prompt *Are you more productive when you are doing a few things at the same time, or are you more productive when you have only a single thing to focus on? What do you think helps you to be more productive?* We would not mark this as a category C shell span because the prompt topic has been paraphrased sufficiently to serve as a topic claim/thesis statement. This distinction is somewhat subjective, and while we achieved good inter-annotator agreement on this category, this point was likely a source of ambiguity for models.

3.4 Category D: Appeal to Authority

This category includes mentions of reports or studies that imply external validation or evidence. Examples include:

- `A report from University of Maryland shows that ...`
- `Oxford University conducted a study that confirmed ...`
- `For example, a report published by The New York Times reveals that ...`

3.5 Category E: Stance-taking

This category contains phrases or statements used to convey the writer's stance or position, whether in the first-person or third. Observed examples include:

- `I feel/believe/think (that) ...`
- `From my point of view ...`
- `In my opinion ...`
- `Yes, I agree with the statement that ...`

The exception for this category are phrases that are used to convey the writer's personal preference and do not serve a stance-taking role. For example,

consider the sentence `I like good environment for touring because i loved with nature`. Here, the phrase `I like` is used to convey the writer's personal preference for a specific type of environment rather than their argumentative stance.

3.6 Rubric usage

Although our final rubric delineates five different categories of shell text, many shell language spans usually serve multiple purposes (e.g., the phrase `More and more people believe that ...` can be said to convey both general emphasis (category B) and the writer's position (category E). In such cases, our practice is to consistently choose the category that seems more relevant in the context of the full response. For purposes of evaluation and error analysis, we also sometimes refer to a sixth category "O" consisting of all spans *not* labeled as shell (the "meat" of the response). We do not separately label this category manually or ask models to directly annotate O spans; it's defined by the absence of annotations for other classes.

Once the final rubric was created, the authors independently annotated 92 additional responses followed by curation, for a final dataset of 142 responses with individual and curated shell annotations. Note that the first 11 annotated responses (used to make major revisions to the initial rubric) are *not* part of this final set, as their annotations do not reflect the final rubric. We then split this dataset into a "training set" of 40 responses, from which few-shot examples are drawn (see §4), and a test set of 102 responses.

4 Scaling Annotation with LLMs

Shell annotation is a complex task and traditional supervised approaches would require a much larger number of annotated examples to train, but human span annotation is time-intensive and tedious. In this section we evaluate the accuracy of annotations elicited with few-shot learning.

4.1 Method

For LLM-based annotation, we compare five models: DeepSeek-V3 (Chandra et al., 1981) GPT-4o (Hurst et al., 2024), DeepSeek-R1 (Liu et al., 2024), o1 (Jaech et al., 2024), and o3-mini (OpenAI, 2025). Note that the first two are non-reasoning models, while the latter three use self-prompting or reasoning techniques recently popularized by o1 and DeepSeek-R1.

<shell category="B"> **In this modern world** </shell>, artificial intelligence <shell category="B"> **is so well known in the world** </shell>, which is a kind of intelligence. <shell category="A"> **Futhermore** </shell>, <shell category="E"> **I firmly agree with this given notion that** </shell> intelligence has distinct types.

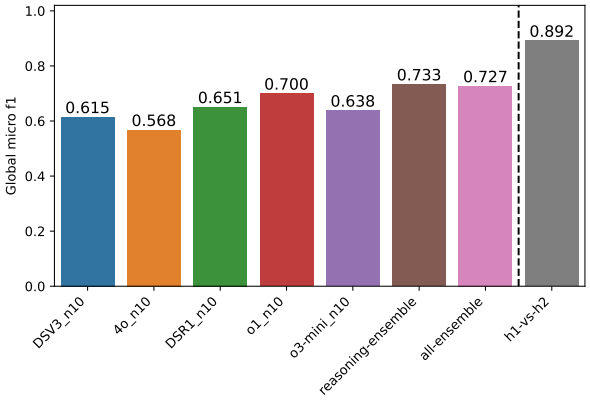
Figure 1: Markup format for LLM annotations.

Model	Success rate	Format errs	Generation errs
DeepSeek-R1	0.75	3	22
DeepSeek-V3	0.95	4	1
gpt-4o	0.93	7	0
o1	0.98	0	2
o3-mini	0.99	0	1

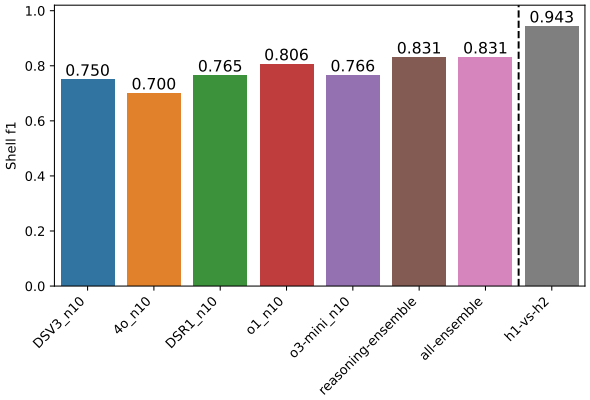
Table 1: Success rate of generating a valid annotation on the first try, by model (using 5 examples). Error counts are out of 102 responses. Note that DeepSeek-R1 had high rates of prematurely truncated responses seemingly unrelated to the task.

For each model, we use the same prompt containing the entire rubric, along with either 5 or 10 example responses annotated in an XML-like format with the shell category as an attribute (see Figure 1). We use this prompt to elicit span annotations on our eval set of 102 instances; the model is provided with the writing prompt and unannotated test-taker response, and responds with the annotated text in the same format as the examples. We chose this format based on its expressivity and convenience and did not experiment with any additional formats for now (see §7 for more discussion).

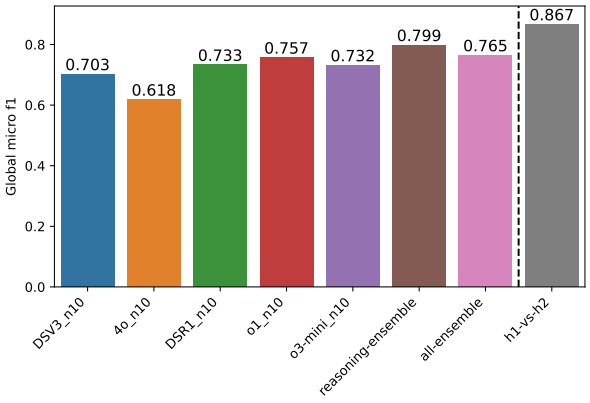
We validate the annotations by trying to automatically parse the XML, checking that there are no nested tags or task-unrelated tags, and that the text is unaltered from the original. The rate of validation failure varies by model. Furthermore, this failure rate is not necessarily constant for a given model; responses where annotation failed on the first round were more likely to also fail on a second try, suggesting that some examples are inherently hard to produce valid annotations for. The most common causes of error were incorrectly formatted XML (unclosed tags, nested tags or non-task-related tags) and missing sentences or phrases. Notably, we found almost no cases where the generated annotated text included unwanted "corrections" of grammatical or typographical er-



(a) Multiclass shell labeling.



(b) Binary shell labeling.



(c) Multiclass shell labeling (B and C categories excluded).

Figure 2: Token-level F_1 for three shell annotation tasks. Reasoning models outperform non-reasoning models, and the ensembles improve slightly over the best individual models. Exclusion of B and C categories improves micro-averaged F_1 for all models.

rors in the original test-taker response, with the exception of whitespace errors such as replacing multiple spaces with a single space or inserting a missing space after sentence-final punctuation. For purposes of evaluation, we automatically resolved these whitespace errors by editing all annotated versions of a text to match the original (to ensure

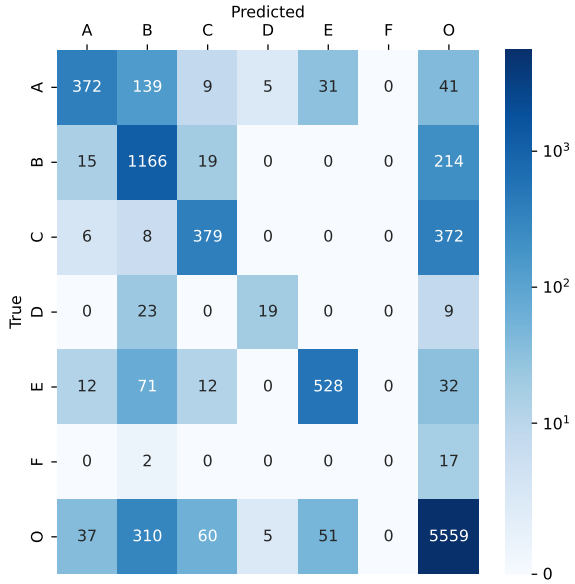


Figure 3: Confusion matrix for `all-ensemble`.

tokenization was compatible) before comparing annotations. For our provider of DeepSeek-R1, a significant fraction of long responses were cut off prematurely, increasing the error rate beyond what was attributable to formatting errors. Table 1 compares the LLMs by the fraction of responses that passed validation with a single request.

4.2 Results

Next, we compute token-level metrics for the LLM annotations using the curated human labels as the gold standard, and compare to the inter-annotator agreement for the human annotators. Figure 2 shows results for binary (shell vs non-shell) and multiclass evaluation.

We present the confusion matrix for the all-model ensemble in Figure 3 as a relatively representative example of the errors made by all models. The counts represent individual token counts. The confusion matrix provides insight into specific pairs of categories often confused; for example, we see that categories D and E have few false positives, i.e. they are rarely predicted when the true label is another category. We also observe that B and C are the categories with the most errors (both because they have high true token counts and because they require the most subjective decisions). For this reason, we also include F_1 results considering only a subset of shell category labels (all except B and C) in Figure 2.

Model	Prompt tokens	Completion Tokens	Total cost
DeepSeek-R1	431,316	127.414	$2.19
DeepSeek-V3	431,737	15,566	$0.56
GPT-4o	430,290	15,859	$1.06
o1	425,061	257,592	$20.28
o3-mini	431,234	492,148	$2.54

Table 2: Comparing prompt tokens, completion tokens, and total cost when annotating our curated evaluation set of 102 responses using various LLMs.

4.3 Costs

Our method of shell annotation with large language models requires a lengthy rubric and several example texts to be provided in the prompt for every instance. This is a relatively costly approach. In addition, our use of reasoning models leads to high completion token counts.

In order to provide a useful comparison of model costs, Table 2 shows prompt/completion token counts and costs when annotating our curated evaluation set of 102 responses using the same set of LLMs we used in §4. Note that we do not retry any incorrectly formatted annotations for this specific set, so the error rates reported in Table 1 should also be considered when comparing these costs.

In addition, one would expect to incur significant upfront costs iterating and validating the annotation scheme and rubrics. In our case, we spent a total of $3,665.45 across all experiments.

5 Supervised Learning to Detect Shell

In this section, we attempt to distill a large number of LLM-based annotations into a BERT variant, ModernBERT (Warner et al., 2024). There are several advantages to this approach: BERT models are cheaper, can be run locally (avoiding dependence on external APIs) and can directly produce per-token labels rather than generating the annotated text, removing a potential source of errors.

Based on the results in §4, we choose OpenAI's o1 model as the best single model for the task. Using the same approach as previously described, we prompt o1 to annotate 7100 additional test-taker responses, and split the resulting dataset into a training set (6500 responses) and a validation set (600 responses). We then convert the annotations into BIO format (Ramshaw and Marcus, 1999) and finetune ModernBERT on three training samples with different sizes: 500, 1000, and the full 6500. For all finetuning runs, we set the batch size to 12 and learning rate to $7e^{-05}$ and train for 10 epochs with early stopping based on the performance on the validation set.[2]

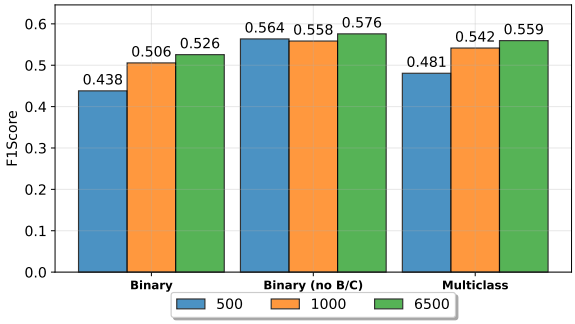


Figure 4: ModernBERT F_1 for each task on the human-labeled test set (102 examples). Notably, multiclass labeling performance is actually higher than binary labeling performance on equal data sizes.

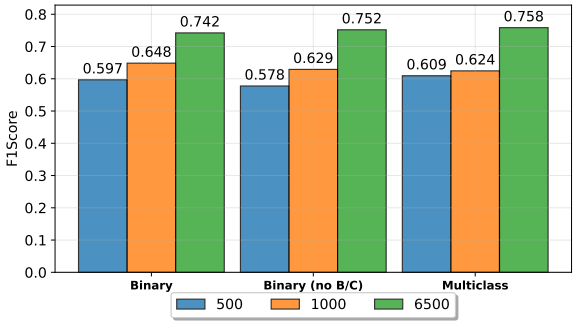


Figure 5: ModernBERT F_1 for each task on the o1-labeled dev set (600 examples). Performance is significantly higher than on the human-labeled test set, suggesting that the BERT model has learned o1-generated patterns that are misaligned to human raters.

We finetune and evaluate ModernBERT on three tasks: binary shell labeling, binary shell labeling with B and C categories excluded, and multiclass labeling (see Figure 4). ModernBERT substantially under-performs the LLM used to train it (o1 with 5 examples) on the human-labeled test set, never surpassing 0.6 F_1 even for the easiest task. However, on the o1-labeled validation set, ModernBERT trained on 6500 examples surpasses 0.75 F1 for multiclass labeling (Figure 5).

[2]Learning rate was chosen based on a search over the validation set when training on 500 responses.

6 Discussion

6.1 o1 error analysis

Shell labeling is a difficult and to some extent subjective task. In this section, we present a qualitative analysis of the differences between the best-performing single LLM (o1) and curated human annotations, along with examples. To improve readability, we use <X>...</X> as a shorthand for <shell category="X">...</shell>.

The most common error categories in the confusion matrix in Figure 3 are missing tokens of categories B (general statements) and C (topic restatement), and various non-B tokens labeled as B. We observe a similar pattern when looking at whole-span errors[3]. The most common cases of whole-span errors are B-spans applied to O and A text. O spans applied to C text (i.e. missed C labels) are also common. This is perhaps to be expected given the rubric; B and C are the most contextual and nuanced categories, requiring consideration of what is specific to the prompt vs generic and what is restatement vs original.

In the following example, the model identified most of the sentence as B, possibly due to the positive emphasis ("outstanding", "wide knowledge") which is often seen in B spans.

Curated O vs. LLM B

<A>First of all</A> <E>it is true that</E> college and university can serve as an outstanding place to gaing wide knowledge and contact as you can meet with like minded individuals

<A>First of all</A> <E>it is true that</E> <B>college and university can serve as an outstanding place to gaing wide knowledge and contact as you can meet with like minded individuals</B>

In another case, a partial reference to the prompt ("the second part of the statement") was mistakenly treated as a restatement of the prompt, as shown below. This may be a case where o1 talked itself into an otherwise unlikely error.

[3]Whole-span errors occur when a predicted span has no overlap with a human-annotated span of the same category. Boundary errors, by contrast, involve partial overlap but incorrect span length.

Curated O vs. LLM C

<E>I storngly prefer</E> the second part of the statement **<A>for many reasons</A>**.

<E>I storngly prefer</E> <C>the second part of the statement</C> <A>for many reasons</A>.

Below we considered this declaration of "heated debate" to be an instance of B, but o1 did not:

Curated B vs. LLM O

<C>intercultural communication can be a valuable learning experience</C> <B>has sparked a heated debate.</B>

<C>intercultural communication can be a valuable learning experience</C> has sparked a heated debate.

A common boundary error involved commas. During manual annotation, we settled on a convention of excluding trailing commas from shell spans but did not explicitly specify this in the rubric. O1 frequently took the opposite approach as shown below, causing a 1-token error.

Curated vs. LLM

<A>To sum up</A>, ...

<A>To sum up,</A> ...

<A>For my experience</A>, ...

<A>For my experience,</A> ...

Finally, we excluded mentions of topics or entities from the prompt from annotations in sentences that were otherwise B. This was attested in a few examples, but not made explicit in the rubric, and o1 tended to include them, leading to boundary errors, as the example shows.

Curated vs. LLM

<B>A serious amount of worldwide attention has been drawn to</B> the intercultural communication. **<B>Beacuse of the existence of evidencen in favour of as well as against the approval of</B>** intercultural communication.

<B>A serious amount of worldwide attention has been drawn to the intercultural communication.</B>

<B>Beacuse of the existence of evidencen in favour of as well as against the approval of intercultural communication.</B>

6.2 Interpretation

Many error types above are consistent and systematic, which is a promising sign for improving the accuracy of automatic shell annotation. In a few cases, o1's annotations were arguably more consistent with the intent of the rubric than the curated human annotations. For example, some spans of A and E were missed by human annotators, categories which were fairly reliably marked by o1. In other cases, o1 marked statements that broadly paraphrased statements from the prompt as C when human annotators judged the paraphrase as original in form, though not content.

As expected, the two step training procedure in §5 results in a model that suffers from two sources of errors: errors between o1 and human annotators, and errors between ModernBERT and o1. In fact, it seems that ModernBERT does not learn to correct any significant portion of o1's errors, as the total error rate is not much better than if the two sources of error were entirely independent: we observe $0.559F_1$ for the largest ModernBERT model for multiclass labeling, vs. 0.531 expected (0.7 o1 F_1 $\times$ 0.752 ModernBERT F1 on o1's labels). This is consistent with LLMs consistently diverging from human annotations; ModernBERT is learning to imitate systematic error, rather than guessing in response to random noise.

7 Related work

The work most closely related to ours and the one we build upon is that of Madnani et al. (2012). However, there are also salient differences between our work and theirs. They rely on a small set of human annotations to train a binary, feature-based, discriminative classifier for shell language whereas we use a small, curated set of human annotations to bootstrap LLM-generated annotations at scale, and then distill them into an end-to-end transformer model used for finer-grained, multi-class, shell span classification. Additionally, while they do not share any information about their annotation process, we share a detailed rubric along with examples for each shell category. Bejar et al. (2013) apply the shell model developed by Madnani et al. (2012) to GRE essays to evaluate whether it agrees with expert raters' judgments and whether the presence of shell language has an effect on the essay scores. Du et al. (2014) devise an unsupervised HMM-LDA topic model for shell language and apply it to posts from online debate forums. Similarly, Ó Séaghdha and Teufel (2014) use a topic model to capture words & constructs used to express rhetorical function in scientific papers.

LLMs have been extensively used for a wide range of linguistic analysis tasks. Some of these tasks are fairly straightforward. For example, Hao et al. (2024) use ChatGPT to annotate conversation chat turns in a collaborative problem solving setting with a pre-defined set of labels. However, the decoder-only framework for text generation makes it difficult to represent more complex linguistic structures such as spans or dependency relations and their relationships to the annotated text, and, to our knowledge, there has been no consensus on the format to use for span annotation with LLMs (regardless of the particular application). Blevins et al. (2022) experimented with LLMs for sequence tagging tasks, including multi-token spans for chunking and NER. They framed the task as BIO tagging at the word level, regenerating the text with labels following each word. Since our spans are frequently even longer than syntactic chunks, and rarely as short as single words, we opt for a format that abstracts away from individual word labels.

More recently, Kasner et al. (2025) experimented with span annotation for evaluation of generated text by using structured decoding to get a list of spans with category labels in JSON format. This has the advantage of not requiring re-generation of the full input text. However, our application does not require full category names for individual annotations (only a single-character label) and we expect to label relatively densely (such that a significant fraction of the text would have to be copied in the output anyway). Future work should directly compare these output formats on a single task (or multiple tasks) and investigate the effects of output format on overall performance and error types.

8 Conclusions and Next Steps

We have shown that LLMs can be used for scalable span annotation and that reasoning models have a distinct advantage at the task of labeling shell text. However, both the original LLM annotation process and training a smaller model to imitate an LLM's annotations remain error-prone. Based on the consistency of certain error types (§6), we believe that refinements to the annotation rubric could significantly improve the accuracy of LLM

annotation. For example, the distinction between a clear restatement of the prompt and its paraphrase is a bit subtle and can be made clearer to ensure a more consistent interpretation. Other directions for future work include:

- a more thorough hyperparameter search to improve supervised learning,
- finetuning a reasoning LLM either directly on the curated human data or a combination of human and LLM-annotated data (given the small size of the human data), and
- experimenting with other output formats from related work such as structured decoding for greater consistency.

While our supervised shell detection results certainly leave room for improvement, we hope that the work done in this paper can still serve as a source of useful information to other researchers working on shell language detection and, more broadly, LLM-based span annotation. We believe that the workflow proposed in this paper can be applied to other types of non-overlapping span-labeling tasks, assuming a rubric with clearly defined categories and reliable human-human agreement.

9 Acknowledgements

We thank the anonymous reviewers, as well as Kevin Yancey and Alina von Davier, for their valuable feedback.

Limitations

There are several limitations of this work. Most significantly, while the two-step annotation procedure we describe yields promising results, the resulting error rate of the final ModernBERT model may limit its application without additional refinements to the rubric and/or the training procedure (including improved hyperparameter tuning). For our LLM experiments, we compared several different models in §4 and found that an ensemble of multiple models performed best. However, our budget and time constraints limited the number of compared models, and, in the end, we had to pick the best single model (o1) to produce training data for ModernBERT instead of the ensemble. Due to limited space, our error analysis only covers errors made by o1, and does not show the extent to which the same patterns may be shared by other LLMs or ModernBERT. Finally, our results are limited to the specific choice of span annotation format that we chose. As mentioned in §7, other formats may have different tradeoffs, which we hope future work will explore.

References

Isaac I. Bejar, Waverely VanWinkle, Nitin Madnani, William Lewis, and Michael Steier. 2013. Length of textual response as a construct-irrelevant response strategy: The case of shell language. *ETS Research Report Series*, 2013(1):i–39.

Terra Blevins, Hila Gonen, and Luke Zettlemoyer. 2022. Prompting language models for linguistic structure. *arXiv preprint arXiv:2211.07830*.

Ashok K. Chandra, Dexter C. Kozen, and Larry J. Stockmeyer. 1981. Alternation. *Journal of the Association for Computing Machinery*, 28(1):114–133.

Jianguang Du, Jing Jiang, Liu Yang, Dandan Song, and Lejian Liao. 2014. Shell miner: Mining organizational phrases in argumentative texts in social media. In *2014 IEEE International Conference on Data Mining*, pages 797–802. IEEE.

Jiangang Hao, Wenju Cui, Patrick Kyllonen, Emily Kerzabi, Lei Liu, and Michael Flor. 2024. Scaling up the evaluation of collaborative problem solving: Promises and challenges of coding chat data with chatgpt. *Preprint*, arXiv:2411.10246.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Zdeněk Kasner, Vilém Zouhar, Patrícia Schmidtová, Ivan Kartáč, Kristỳna Onderková, Ondřej Plátek, Dimitra Gkatzia, Saad Mahamood, Ondřej Dušek, and Simone Balloccu. 2025. Large language models as span annotators. *arXiv preprint arXiv:2504.08697*.

Jan-Christoph Klie, Michael Bugert, Beto Boullosa, Richard Eckart de Castilho, and Iryna Gurevych. 2018. The inception platform: Machine-assisted and knowledge-oriented interactive annotation. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 5–9. Association for Computational Linguistics. Event Title: The 27th International Conference on Computational Linguistics (COLING 2018).

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Nitin Madnani, Michael Heilman, Joel Tetreault, and Martin Chodorow. 2012. Identifying high-level organizational elements in argumentative discourse. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 20–28, Montréal, Canada. Association for Computational Linguistics.

Diarmuid Ó Séaghdha and Simone Teufel. 2014. Unsupervised learning of rhetorical structure with un-topic models. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2–13, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

OpenAI. 2025. Openai o3-mini technical report. Technical report.

Lance A Ramshaw and Mitchell P Marcus. 1999. Text chunking using transformation-based learning. In *Natural language processing using very large corpora*, pages 157–176. Springer.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Nathan Cooper, Griffin Adams, Jeremy Howard, and Iacopo Poli. 2024. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. *Preprint*, arXiv:2412.13663.