Comparative Analysis of BERT & ModernBERT in Lexical Complexity Classification

Jonathan Hernandez
jonhernandez@berkeley.edu

School of Information, University of California, Berkeley

Abstract

In this work, we introduce a binarized approach to Lexical Complexity Prediction (Binary LCP) and systematically compare two generations of encoder-only Transformer models: BERT and ModernBERT. Building on the SemEval-2021 LCP dataset, we re-label continuous complexity scores as complex vs. not-complex and leverage data enrichment methods involving morphosyntactic features. Our experiments reveal that ModernBERT often outperforms BERT in Recall and F1 across multiple domains, with further gains observed when morphological features are appended and interleaved in the input. Our error analysis concludes that domain-specific vocabulary remains a key challenge for European Parliament and Bible subcorpora, while Biomed spans perform better, and exhibit smaller KL Divergence distances in embedded space across models. Moreover, ModernBERT's average token embeddings of predicted classes reveal tighter clustering than BERT, despite differences in embedding scales and vocabularies. Our findings suggest that modernized architectures with larger pre-training corpora and careful data augmentation can substantially advance binarized lexical complexity classification.

1. Introduction

In the last ten years, Complex Word Identification (CWI) and Lexical Complexity Prediction (LCP) tasks have become intrinsically connected to the task of Lexical Simplification (LS), which has the goal to replace complex words and expressions with simpler alternatives, in order to improve outcomes for readers. Lexical complexity tasks are crucial in reading assistance, and help make reading more accessible for a wide variety of readers. Between 2014 and 2021, researchers addressed multi-domain Multi-Word Expressions (MWEs) with a variety of classification and regression techniques, addressing individual words or spans with a variety of systems [1-3].

In the last eight years, Transformer models [4] have helped NLP move from predominantly rules-based, probabilistic, and sequence transduction models, to contextual attention-based representational (encoder) and generative (decoder) models. Then, Bidirectional Transformers for Language Understanding [5] brought about cleverly-trained BERT models, with open pre-trained weights—allowing practitioners to fine-tune them for domains and downstream tasks.

Recently, ModernBERT [6], a new encoder-only model family updates the original BERT approach with a modernized architecture, and a comprehensive 2 trillion token dataset. Since then, a small study [7] comparing both models' F1 score performance on the same dataset, found that ModernBERT outperformed BERT, the majority of the time, on classification tasks.

Given these developments, we are inspired toward our objective, to compare the performance and error patterns of BERT and ModernBERT on a newly binarized Lexical Complexity dataset, using F1 and Precision/Recall as core metrics—and to investigate whether the updated architecture of ModernBERT yields systematically better classification outcomes for MWEs in lexical complexity tasks. We call this task, Binary LCP.

2. Background

We adopt the SemEval-2021 LCP dataset [3], and convert their continuous labels into binary classification labels. We then fine-tune the top layers of BERT models with ablations of raw and enriched MWEs, whose inherent variation in tokenized embedding values alllows us to stochastically test model performance with the same data, in raw and transformed states.

This process is aligned with historical CWI tasks [1-2], and is necessary because no pubhished work has systematically benchmarked BERT vs. ModernBERT architectures on binary LCP tasks. In support of our aims, we employ novel and reliable techniques to perform the data conversion, which allows us to preserve both models' [5-6] native architecture and pre-trained classification heads. Had we performed regression-based LCP [3], we would need to remove the top model layers and concatenate their attention output with the pooled output of the encoder, before passing them to a regression model [8].

The remainder of this paper is structured as follows: Section 3 describes the Dataset, Section 4 details our Methods, Models, and Baselines, Section 5 provides quantitative Results and Discussion, and Section 6 includes our Conclusion.

3 Dataset

The SemEval 2021's LCP dataset provides 10,800 multi-word spans drawn from three domains, the European Parliament, Biomed research, and the Bible. Each record was labeled and annotated with a lexical complexity score within a continuous range. The labels served two tasks: 1) Single Word Complexity (Task 1): Each of the 7,662 instances contain an MWE where one token was chosen from within the MWE to derive its complexity score in the range [0,1]; and 2) Multi-Word Expression (MWE) Complexity (Task 2): Each of the 1,800 instances contain an MWE where a bigram of two tokens were chosen from within the MWE to derive its complexity score.

3.1 Dataset Composition

The original LCP labels were likert-scale-based averages, reflecting the degree of difficulty that human annotators perceived for each target token (unigram or bigram) within the MWE. Each token was present in the dataset more than once, and at most five times. The dataset was composed with an 85/5/10 split, between train, validation (dev), and test. In addition, the dataset was highly consistent between tasks, such that median complexity values between data splits were always within 0.02 absolute values of each other. Task 1's Single set had a median complexity value of 0.2733 and Task 2's Multi set had a median value of 0.42, indicating divergence in annotator's LCP perspectives.

Quartiles	Single-Task Set		
	Train	Validation	Test
Q1 - Q2: Less Complex	3,865	229	476
Q3 - Q4: More Complex	3,797	192	441
Total:	7,662	421	917

Quartiles	Multi-Task Set		
	Train	Validation	Test
Q1 - Q2: Less Complex	759	51	99
Q3 - Q4: More Complex	758	48	85
Total:	1517	99	184

Table 1: Records by Quartile (Original Split of Continuous LCP Outcome Variable)

3.2 Data Engineering

Given that the top-performing teams for the original tasks [3] used data enrichment techniques [8-10], we pursued the same strategy, with the intent to provide our models with comprehensive ablations of the source data.

As shown in Tables 1 and 2, we began by binarizing the outcome variable, by splitting our data on the median of the original continuous complexity outcome variable. We created a separate outcome variable split on the 75th percentile as well, in order to test our assumptions for regressions. Then, we created two sets of the single and multi data, one was re-balanced and one retained the original [3] data split. We did this to increase the size of the validation set so it was equivalent with the test set, and used controls to ensure no subcorpus became over-represented in any split.

Quartiles	Single-Task Set		
	Train	Validation	Test
Q1 - Q2: Not Complex	3,534	229	476
Q3 - Q4: Complex	3,466	192	441
Total:	7,000	1,000	1,000

Quartiles	Multi-Task Set		
	Train	Validation	Test
Q1 - Q2: Not Complex	665	518	99
Q3 - Q4: Complex	635	482	85
Total:	1300	250	250

Table 2: Records by Quartile (Re-balanced Split of Binarized LCP Outcome Variable)

After calculating maximum and average span lengths, to ensure spans would mostly fit in our target models' context windows, we removed contractions [8, 11], used spaCy to generate 4 new representations of our input sequences (i.e. Part-of-Speech, Token Dependency, Morphological Strings, and an Average Morphological Complexity), and 7 new features that both appended or interleaved these representations with our input sequences.

In total, we started with 1 dataset with 2 tasks and rebalanced it into 2 datasets with the same 2 tasks. We then created 2 outcome variables split on the 2nd and 3rd quartiles, and engineered 11 new input features—totaling 13 new features per task, or 26 per dataset. Examples are included in Appendix B.

4. Methods

4.1 Models

In executing this research, we systematically tested BERT-base-cased, BERT-large-cased, ModernBERT-base, and ModernBERT-large [5-6]—we leveraged Naive Bayes as our Baseline Model [13]. Results were evaluated predominantly with F1 Scores, while leveraging Precision and Recall for explainability.

4.2 Baseline Models

Multinomial Naive Bayes is a lightweight classifier that we used to evaluate combinations of our X and Y variables, with Precision, Recall, and F1 Scores. We established baseline performance for these combinations by predicting the binary outcome variable on the training sets for Tasks 1 and 2. We accomplish this by simply predicting the majority class (not-complex) for every sample.

4.3 Transformers Models

4.3.1 BERT, base and large

We use the encoder-only 12-layer BERT-Base and 24-layer BERT-Large models, with 110M and 340M parameters respectively, and a maximum sequence length of 512 tokens. They were pre-trained with a 15% masked language modeling (MLM task on the BooksCorpus and Wikipedia). For fine-tuning, we used the WordPiece tokenizer, gelu activation function and modified the weights of only the top, pooler, and classification layers [5].

4.3.2 ModernBERT, base and large

We additionally use the encoder-only 22-layer ModernBERT-Base and 28-layer ModernBERT-Large models, with 149M and 395M parameters respectively, and a maximum sequence length of 8,192 tokens [6]. They were pre-trained on 2T tokens on mixed domains, with a 30% MLM task. For fine-tuning, we used the Byte-pair encoding tokenizer, gated gelu activation function, and modified the weights of only the top, pooler, and classification layers.

4.3.3 Hyperparameter Tuning

Consistent with the literature, we train with warm up steps configured, testing 5%, 10%, and 100% of steps per configured batch size [14]. In addition, we evaluated multiple parameters in order to standardize a configuration across all models and data ablations. This process included building in automated training of DeBERTa V3, XLNet, and RoBERTa into our pipeline, to ascertain the stability of our results. Our standard configuration that consistently outperformed baseline results consisted of 1 epoch, 1e-5 learning rate, 0.5 weight decay (regularization), attention and hidden dropout defaults, and our median-split binary complexity (over than the 3rd quartile split).

5. Results & Discussion

5.1 Results

Appendix C represents the F1, Precision, Recall, and Accuracy for our Naive Bayes Baseline model on both Single (Task 1) and Multi (Task 2) sets. We highlight several observations:

1) Re-balanced data consistently outperforms the original data split.

2) SpaCy-derived engineered features consistently outperform the raw features.

3) Given the dataset's intentionally complex design, average performance suffered, while top-line performance often led to overfitting.

4) ModernBERT-base consistently outperformed Naive Bayes, BERT-base, BERT-large, and even ModernBERT-large

As seen in Tables 3, 4, and Appendix D, models demonstrated excellent Recall. In addition, multiple experiment configurations (of data ablations, parameters and models) achieved the exact same Validation and Test F1 Scores—which we discovered often occurred when perfect Precision or Recall occurred in the Training and Validation sets.

Typically, this overfitting signal would propagate throughout sets if seen early, but was not tied to a particular data split or X Variable. Though, we did note that perfect Recall occurred more often than perfect Precision, indicating our data made it easier for language models to predict all actual positives present in our data, rather than correctly predict true positives consistently. In addition, performance was consistently higher on the Multi dataset.

Model:	Precision	Recall	F1
bert-base-cased	0.45938	0.53489	0.42765
bert-large-cased	**0.50057**	0.35012	0.35256
modernbert-base-cased	0.49728	**0.94757**	**0.64800**
modernbert-large-cased	0.47205	0.57760	0.50285

Table 3: Per Model Average Performance (Both Tasks)

Model:	Precision	Recall	F1
bert-base-cased	-20.93463	-7.247872	-25.221129
bert-large-cased	**-14.07775**	-39.80565	-38.31418
modernbert-base-cased	-14.09739	**65.04024**	**13.90483**
modernbert-large-cased	-17.11845	2.10522	-10.17257

Table 4: Per Model Average Performance *Relative to Baseline*

5.2 Error Analysis

ModernBERT-base-cased exhibsits the highest recall overall (0.94757), but at a precision of only 0.49728. This pattern suggests that ModernBERT-base correctly identifies the vast majority of actual complex items (leading to near-perfect recall), yet it also over-predicts “complex” labels. This over-prediction often arises when the model latches onto morphosyntactic cues in the data enrichment features, yielding more false positives. Nonetheless, ModernBERT-base’s net F1 across tasks remains strong (0.64800), with a gain of +13.90 relative to baseline. By contrast BERT-large-cased and BERT-base cased strikes a moderately balanced trade-off, though they still underperform both the baseline and ModernBERT-base in F1. Surprisingly, ModernBERT-large-cased yields lower net F1 gains than ModernBERT-base, suggesting that merely relying on scaling up the model size does not guarantee improved performance on a relatively small or idiosyncratic dataset.

Upon analysis of the KL divergence of the average embeddings of correct and incorrect model predictions, we discovered that the European Parliament and Bible subcorpora were consistently more diverged in the embedded space. Moreover, ModernBERT-large did not, on average, outperform ModernBERT-base.

6. Conclusion

In this paper, we proposed a Binary Lexical Complexity Prediction (Binary LCP) framework by converting continuous LCP labels [3] into binary classes for complex vs. not-complex. We systematically compared two generations of encoder-only Transformer models, BERT and ModernBERT, under various data-enrichment conditions. Our experiments highlighted:

1) ModernBERT consistently outperformed BERT across multiple domains (European Parliament, Biomed, and Bible).

2) Morphosyntactic data enrichment improves performance.

3) Domain-Specific Vocabulary remains challenging, notably for the European

Parliament and Bible subcorpora, while Biomed showed significantly tighter embedding clusters and often smaller KL divergence between class predictions.
4) Despite our heavy use of regularization, we still observed instances of perfect recall and identical validation and test scores in certain experimental splits, this suggests that future work might explore domain adaptation strategies, such as multi-stage fine-tuning for specific subcorpora, then evaluating against the full dataset.
5) Fine-tuning the top model layers consistently achieved the highest performance, while fine-tuning embeddings and bottom layers would achieve near-baseline performance.
6) Since our binarized approach intentionally simplifies the continuous LCP task, future research could explore multi-channel outputs from these encoder-only models into fine-tuned regression and classification models.

Overall, our findings underscore that modern Transformer architectures, combined with data enrichment to integrate morphological features, hold promise for improving lexical complexity identification, and by extension, assisting with real-world text simplification tasks.

Acknowledgements
We thank Mark Butler, Natalie Ahn, James Kunz, Joachim Rahmfeld, and Rachel Gao for their consistently empowering, and good-natured, leadership.

References

[1] Gustavo Paetzold and Lucia Specia, 2016 Task 11: Complex Word Identification

[2] Sid Yimam, Chris Biemann, Shervin Malmasi, Gustavo Paetzold, Lucia Specia, Sanjana Stajner, Anais Tack, Marcos Zampieri, 2018 A Report on the Complex Word Identification Shared Task

[3] Matthew Shardlow, Richard Evans, Gustavo Henrique Paetzold, Marcos Zampieri SemEval-2021 Task 1: Lexical Complexity Prediction

[4] Aashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Lion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polusukhin, 2017 Attention Is All You Need

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, 2018 BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

[6] Benjamin Warner, Antoine Chaffin, Benjamin Clavie, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Nathan Cooper, Griffin Adams, Jeremy Howard, Iacopo Poli, 2024 *Smarter, Better Faster, Longer*: A Modern Bidirectional Encoder for Fast, Memory Efficient, and Long Conext Finetuning and Inference

[7] Yosuke Yamagishi, Tomohiro Kikuchi, Souhei Hanaoka, Takehaur Yoshiawa, Osamu Abe, 2025 ModernBERT is More Efficient than Conventional BERT for Chest CT Findings Classification in Japanese Radiology Reports

[8] Nabil El Mamoun, Abdelkader El Mahdaouy, Abdellah El Mekki, Kabil Essefar, Ismail Berrada, 2021 CS-UM6P at SemEval-2021 Task 1: A Deep Learning Model-based Pre-trained Transformer Encoder for Lexical Complexity

[9] Yuki Taya, Lis Kanashiro Pereira, Fei Cheng, Ichiro Kobayashi, 2021 OCHADAI-KYOTO at SemEval-2021 Task 1: Enhancing Model Generalization and Robustness for Lexical Complexity Prediction

[10] Chunguang Pan, Bingyan Song, Shengguang Wang, Zhipeng Luo, 2021 DeepBlueAI at SemEval-2021 Task 1: Lexical Complexity Prediction with A Deep Ensemble Approach

[11] Pascal van Kooten, 2021 Contractions

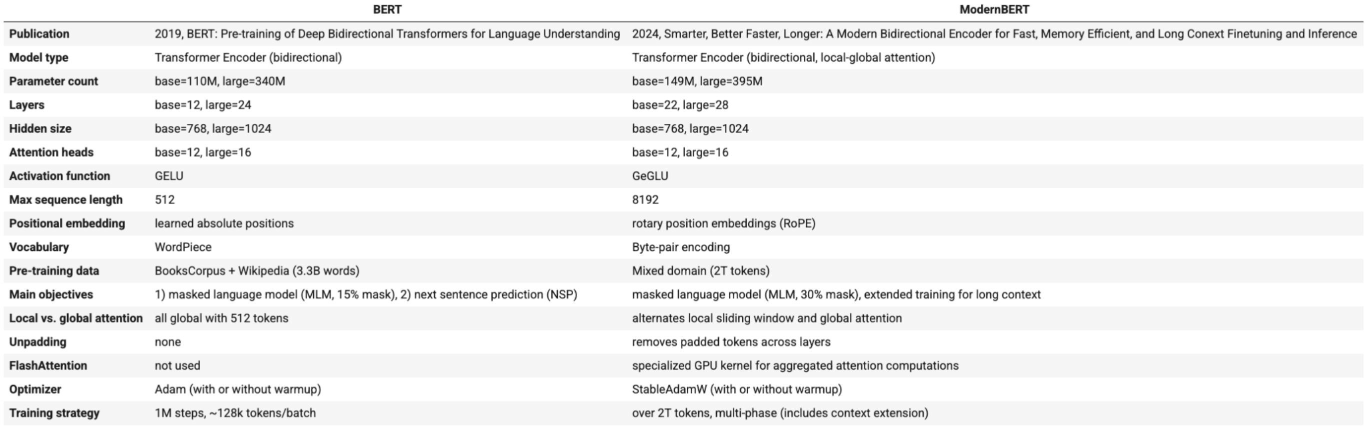
[12] ExplosionAI GmbH, spaCy GmbH, Matthew Honnibal, 2016-2024 spaCy

[13] Vijaykumar B, Vikramkumar, Trilochan, 2014 Bayes and Naive-Bayes Classifier

[14] Marius Mosbach, Maksym Andriuschenko, Dietrich Klakow, 2021 On the Stability of Fine-Tuning BERT: Misconceptions, Explanations, and Strong Baselines (v3)

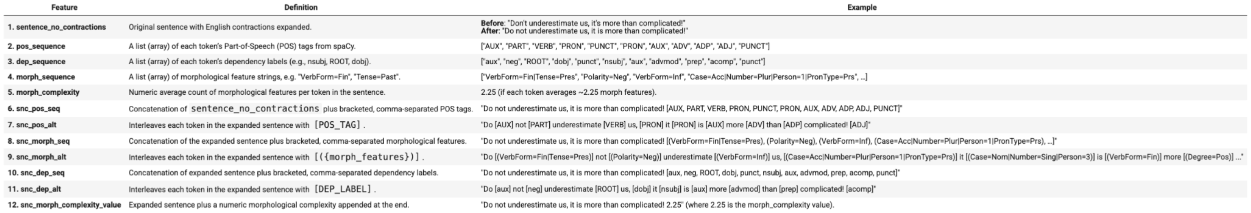
Appendix

Appendix A: BERT & ModernBERT Technical Comparison



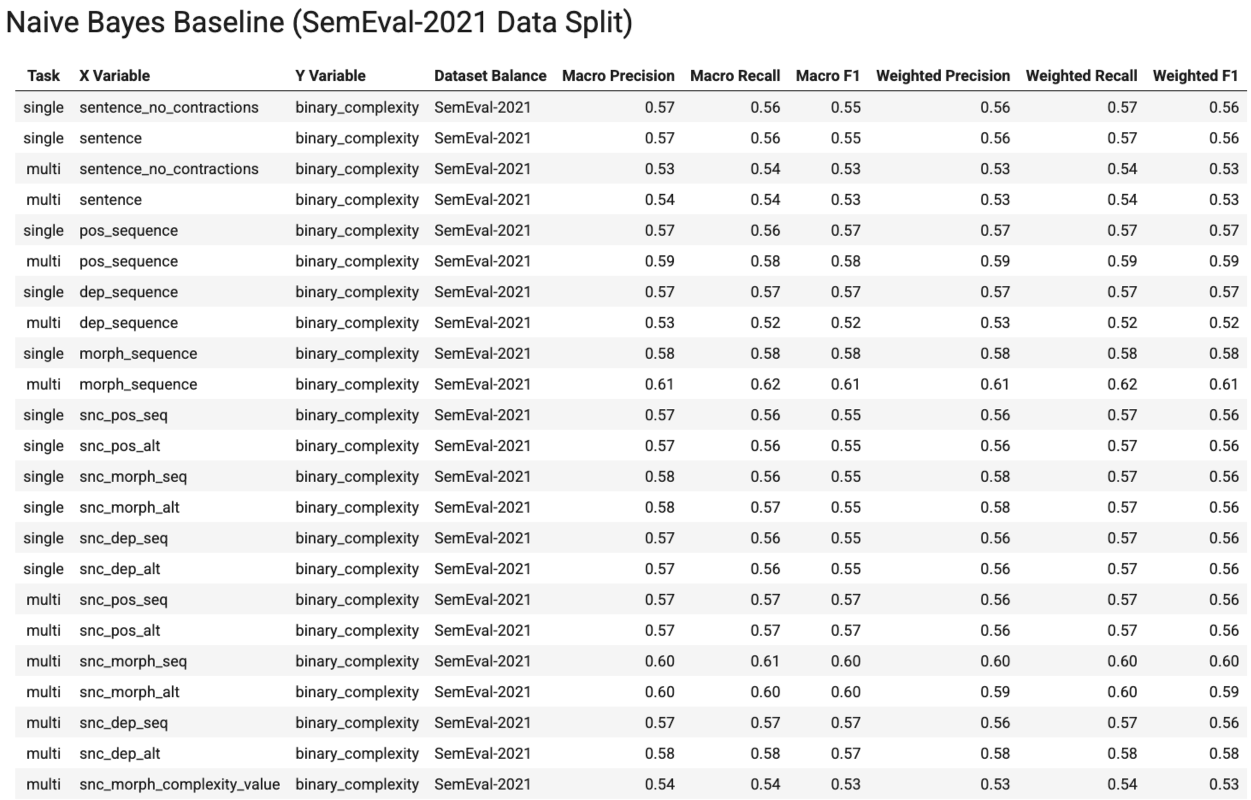
	BERT	ModernBERT
Publication	2019, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding	2024, Smarter, Better Faster, Longer: A Modern Bidirectional Encoder for Fast, Memory Efficient, and Long Conext Finetuning and Inference
Model type	Transformer Encoder (bidirectional)	Transformer Encoder (bidirectional, local-global attention)
Parameter count	base=110M, large=340M	base=149M, large=395M
Layers	base=12, large=24	base=22, large=28
Hidden size	base=768, large=1024	base=768, large=1024
Attention heads	base=12, large=16	base=12, large=16
Activation function	GELU	GeGLU
Max sequence length	512	8192
Positional embedding	learned absolute positions	rotary position embeddings (RoPE)
Vocabulary	WordPiece	Byte-pair encoding
Pre-training data	BooksCorpus + Wikipedia (3.3B words)	Mixed domain (2T tokens)
Main objectives	1) masked language model (MLM, 15% mask), 2) next sentence prediction (NSP)	masked language model (MLM, 30% mask), extended training for long context
Local vs. global attention	all global with 512 tokens	alternates local sliding window and global attention
Unpadding	none	removes padded tokens across layers
FlashAttention	not used	specialized GPU kernel for aggregated attention computations
Optimizer	Adam (with or without warmup)	StableAdamW (with or without warmup)
Training strategy	1M steps, ~128k tokens/batch	over 2T tokens, multi-phase (includes context extension)

Appendix B: Feature Engineering Examples



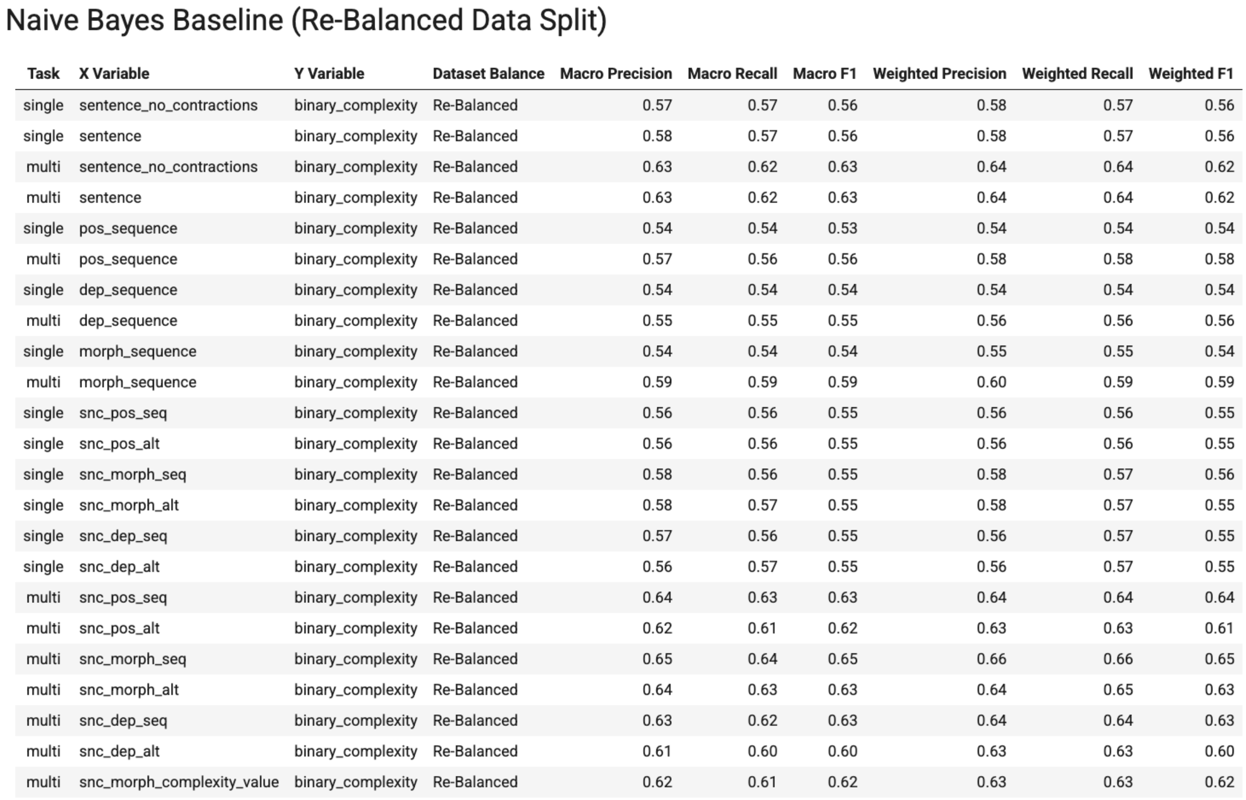
Feature	Definition	Example
1. sentence_no_contractions	Original sentence with English contractions expanded.	**Before**: "Don't underestimate us, it's more than complicated!" **After**: "Do not underestimate us, it is more than complicated!"
2. pos_sequence	A list (array) of each token's Part-of-Speech (POS) tags from spaCy.	["AUX", "PART", "VERB", "PRON", "PUNCT", "PRON", "AUX", "ADV", "ADP", "ADJ", "PUNCT"]
3. dep_sequence	A list (array) of each token's dependency labels (e.g., nsubj, ROOT, dobj).	["aux", "neg", "ROOT", "dobj", "punct", "nsubj", "aux", "advmod", "prep", "acomp", "punct"]
4. morph_sequence	A list (array) of morphological feature strings, e.g. "VerbForm=Fin", "Tense=Past".	["VerbForm=Fin|Tense=Pres", "Polarity=Neg", "VerbForm=Inf", "Case=Acc|Number=Plur|Person=1|PronType=Prs", ...]
5. morph_complexity	Numeric average count of morphological features per token in the sentence.	2.25 (if each token averages ~2.25 morph features).
6. snc_pos_seq	Concatenation of `sentence_no_contractions` plus bracketed, comma-separated POS tags.	"Do not underestimate us, it is more than complicated! [AUX, PART, VERB, PRON, PUNCT, PRON, AUX, ADV, ADP, ADJ, PUNCT]"
7. snc_pos_alt	Interleaves each token in the expanded sentence with `[POS_TAG]` .	"Do [AUX] not [PART] underestimate [VERB] us, [PRON] it [PRON] is [AUX] more [ADV] than [ADP] complicated! [ADJ]"
8. snc_morph_seq	Concatenation of the expanded sentence plus bracketed, comma-separated morphological features.	"Do not underestimate us, it is more than complicated! [(VerbForm=Fin|Tense=Pres), (Polarity=Neg), (VerbForm=Inf), (Case=Acc|Number=Plur|Person=1|PronType=Prs), ...]"
9. snc_morph_alt	Interleaves each token in the expanded sentence with `[({morph_features})]` .	"Do [(VerbForm=Fin|Tense=Pres)] not [(Polarity=Neg)] underestimate [(VerbForm=Inf)] us, [(Case=Acc|Number=Plur|Person=1|PronType=Prs)] it [(Case=Nom|Number=Sing|Person=3)] is [(VerbForm=Fin)] more [(Degree=Pos)] ..."
10. snc_dep_seq	Concatenation of expanded sentence plus bracketed, comma-separated dependency labels.	"Do not underestimate us, it is more than complicated! [aux, neg, ROOT, dobj, punct, nsubj, aux, advmod, prep, acomp, punct]"
11. snc_dep_alt	Interleaves each token in the expanded sentence with `[DEP_LABEL]` .	"Do [aux] not [neg] underestimate [ROOT] us, [dobj] it [nsubj] is [aux] more [advmod] than [prep] complicated! [acomp]"
12. snc_morph_complexity_value	Expanded sentence plus a numeric morphological complexity appended at the end.	"Do not underestimate us, it is more than complicated! 2.25" (where 2.25 is the morph_complexity value).

Appendix C: Baseline Model Results



Naive Bayes Baseline (SemEval-2021 Data Split)

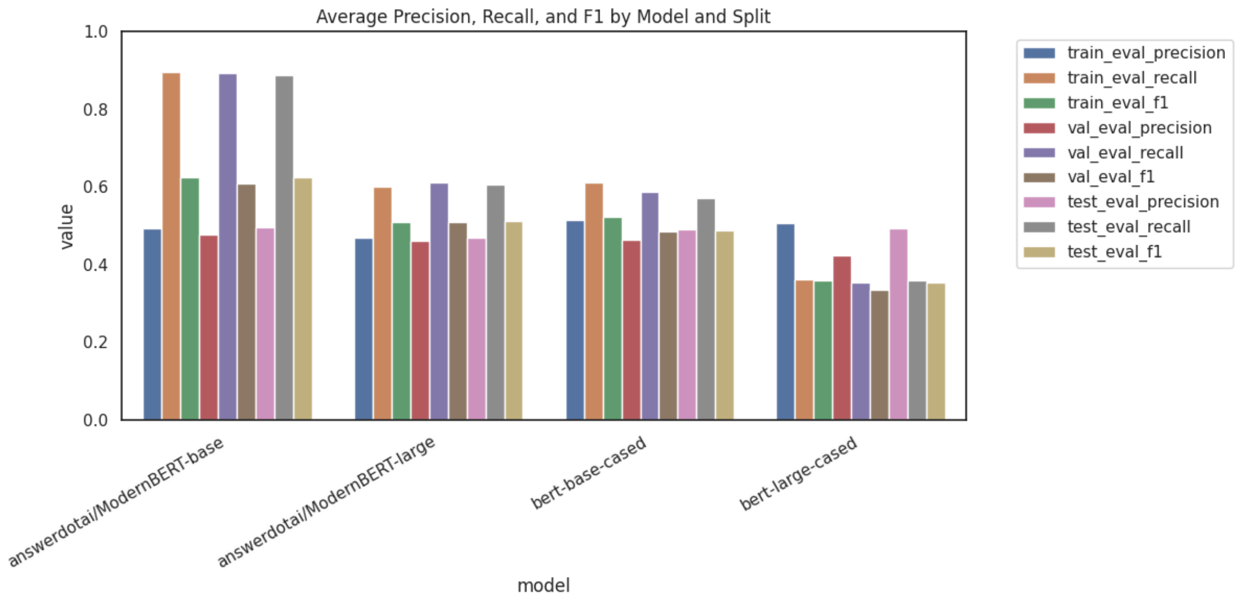
Task	X Variable	Y Variable	Dataset Balance	Macro Precision	Macro Recall	Macro F1	Weighted Precision	Weighted Recall	Weighted F1
single	sentence_no_contractions	binary_complexity	SemEval-2021	0.57	0.56	0.55	0.56	0.57	0.56
single	sentence	binary_complexity	SemEval-2021	0.57	0.56	0.55	0.56	0.57	0.56
multi	sentence_no_contractions	binary_complexity	SemEval-2021	0.53	0.54	0.53	0.53	0.54	0.53
multi	sentence	binary_complexity	SemEval-2021	0.54	0.54	0.53	0.53	0.54	0.53
single	pos_sequence	binary_complexity	SemEval-2021	0.57	0.56	0.57	0.57	0.57	0.57
multi	pos_sequence	binary_complexity	SemEval-2021	0.59	0.58	0.58	0.59	0.59	0.59
single	dep_sequence	binary_complexity	SemEval-2021	0.57	0.57	0.57	0.57	0.57	0.57
multi	dep_sequence	binary_complexity	SemEval-2021	0.53	0.52	0.52	0.53	0.52	0.52
single	morph_sequence	binary_complexity	SemEval-2021	0.58	0.58	0.58	0.58	0.58	0.58
multi	morph_sequence	binary_complexity	SemEval-2021	0.61	0.62	0.61	0.61	0.62	0.61
single	snc_pos_seq	binary_complexity	SemEval-2021	0.57	0.56	0.55	0.56	0.57	0.56
single	snc_pos_alt	binary_complexity	SemEval-2021	0.57	0.56	0.55	0.56	0.57	0.56
single	snc_morph_seq	binary_complexity	SemEval-2021	0.58	0.56	0.55	0.58	0.57	0.56
single	snc_morph_alt	binary_complexity	SemEval-2021	0.58	0.57	0.55	0.58	0.57	0.56
single	snc_dep_seq	binary_complexity	SemEval-2021	0.57	0.56	0.55	0.56	0.57	0.56
single	snc_dep_alt	binary_complexity	SemEval-2021	0.57	0.56	0.55	0.56	0.57	0.56
multi	snc_pos_seq	binary_complexity	SemEval-2021	0.57	0.57	0.57	0.56	0.57	0.56
multi	snc_pos_alt	binary_complexity	SemEval-2021	0.57	0.57	0.57	0.56	0.57	0.56
multi	snc_morph_seq	binary_complexity	SemEval-2021	0.60	0.61	0.60	0.60	0.60	0.60
multi	snc_morph_alt	binary_complexity	SemEval-2021	0.60	0.60	0.60	0.59	0.60	0.59
multi	snc_dep_seq	binary_complexity	SemEval-2021	0.57	0.57	0.57	0.56	0.57	0.56
multi	snc_dep_alt	binary_complexity	SemEval-2021	0.58	0.58	0.57	0.58	0.58	0.58
multi	snc_morph_complexity_value	binary_complexity	SemEval-2021	0.54	0.54	0.53	0.53	0.54	0.53



Naive Bayes Baseline (Re-Balanced Data Split)

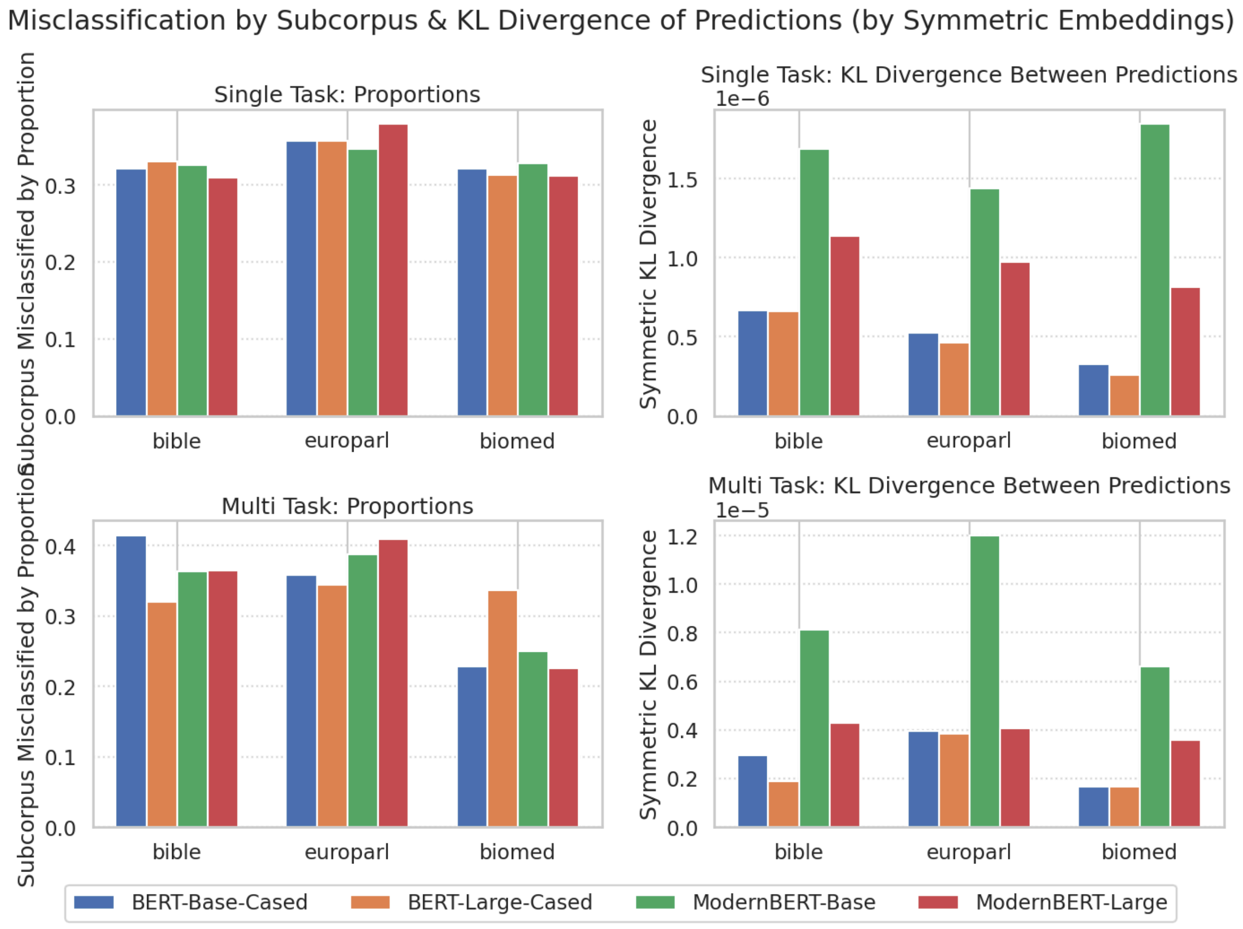
Task	X Variable	Y Variable	Dataset Balance	Macro Precision	Macro Recall	Macro F1	Weighted Precision	Weighted Recall	Weighted F1
single	sentence_no_contractions	binary_complexity	Re-Balanced	0.57	0.57	0.56	0.58	0.57	0.56
single	sentence	binary_complexity	Re-Balanced	0.58	0.57	0.56	0.58	0.57	0.56
multi	sentence_no_contractions	binary_complexity	Re-Balanced	0.63	0.62	0.63	0.64	0.64	0.62
multi	sentence	binary_complexity	Re-Balanced	0.63	0.62	0.63	0.64	0.64	0.62
single	pos_sequence	binary_complexity	Re-Balanced	0.54	0.54	0.53	0.54	0.54	0.54
multi	pos_sequence	binary_complexity	Re-Balanced	0.57	0.56	0.56	0.58	0.58	0.58
single	dep_sequence	binary_complexity	Re-Balanced	0.54	0.54	0.54	0.54	0.54	0.54
multi	dep_sequence	binary_complexity	Re-Balanced	0.55	0.55	0.55	0.56	0.56	0.56
single	morph_sequence	binary_complexity	Re-Balanced	0.54	0.54	0.54	0.55	0.55	0.54
multi	morph_sequence	binary_complexity	Re-Balanced	0.59	0.59	0.59	0.60	0.59	0.59
single	snc_pos_seq	binary_complexity	Re-Balanced	0.56	0.56	0.55	0.56	0.56	0.55
single	snc_pos_alt	binary_complexity	Re-Balanced	0.56	0.56	0.55	0.56	0.56	0.55
single	snc_morph_seq	binary_complexity	Re-Balanced	0.58	0.56	0.55	0.58	0.57	0.56
single	snc_morph_alt	binary_complexity	Re-Balanced	0.58	0.57	0.55	0.58	0.57	0.55
single	snc_dep_seq	binary_complexity	Re-Balanced	0.57	0.56	0.55	0.56	0.57	0.55
single	snc_dep_alt	binary_complexity	Re-Balanced	0.56	0.57	0.55	0.56	0.57	0.55
multi	snc_pos_seq	binary_complexity	Re-Balanced	0.64	0.63	0.63	0.64	0.64	0.64
multi	snc_pos_alt	binary_complexity	Re-Balanced	0.62	0.61	0.62	0.63	0.63	0.61
multi	snc_morph_seq	binary_complexity	Re-Balanced	0.65	0.64	0.65	0.66	0.66	0.65
multi	snc_morph_alt	binary_complexity	Re-Balanced	0.64	0.63	0.63	0.64	0.65	0.63
multi	snc_dep_seq	binary_complexity	Re-Balanced	0.63	0.62	0.63	0.64	0.64	0.63
multi	snc_dep_alt	binary_complexity	Re-Balanced	0.61	0.60	0.60	0.63	0.63	0.60
multi	snc_morph_complexity_value	binary_complexity	Re-Balanced	0.62	0.61	0.62	0.63	0.63	0.62

Appendix D: Average Precision, Recall, and F1 by BERT Models and Data Split



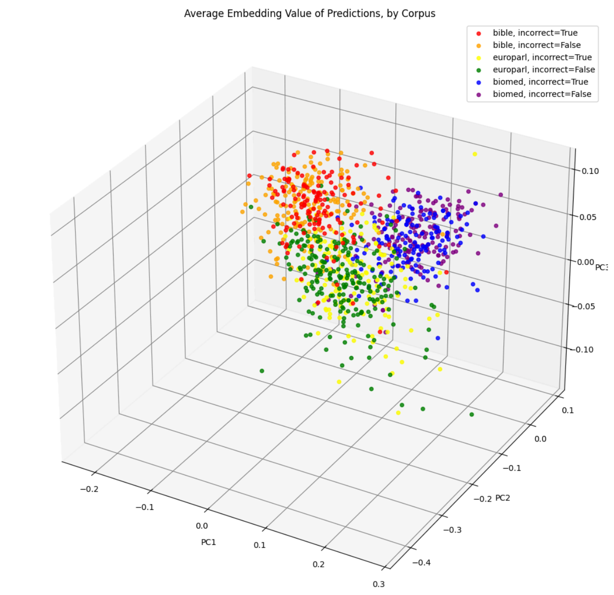
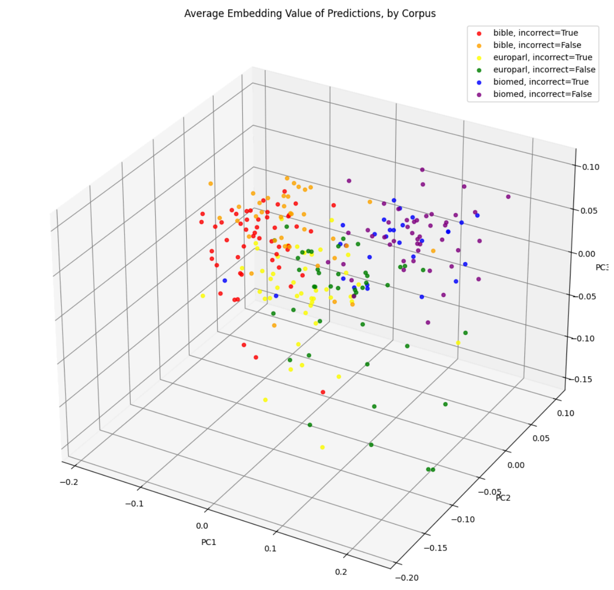
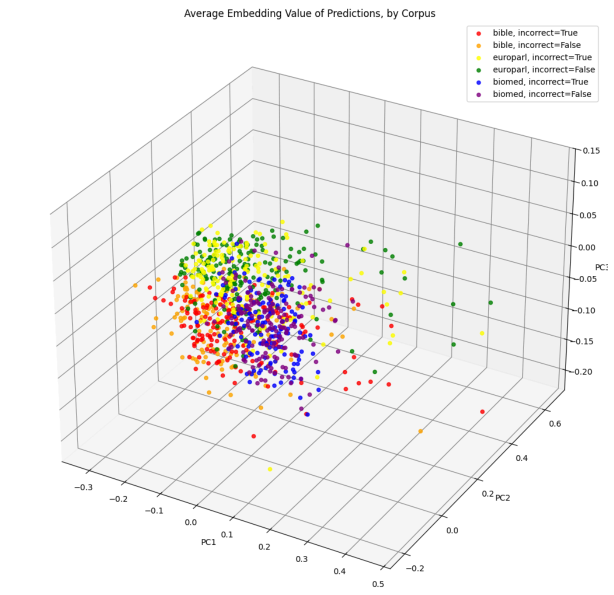
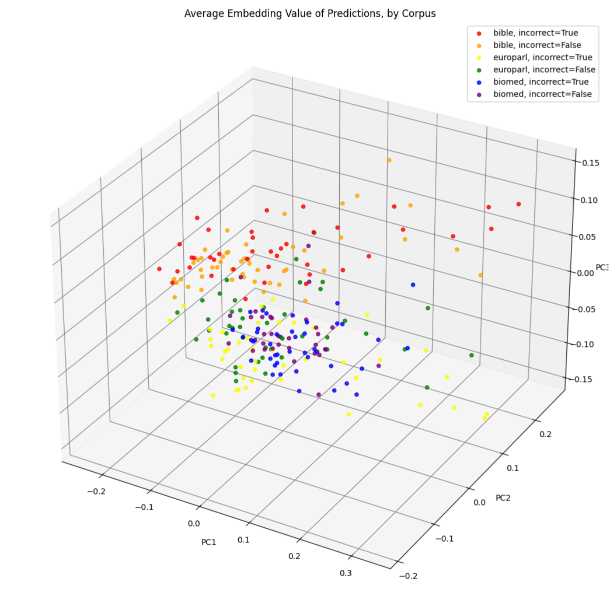
Appendix E: Misclassification Errors by Subcorpus & KL Divergence of Predictions

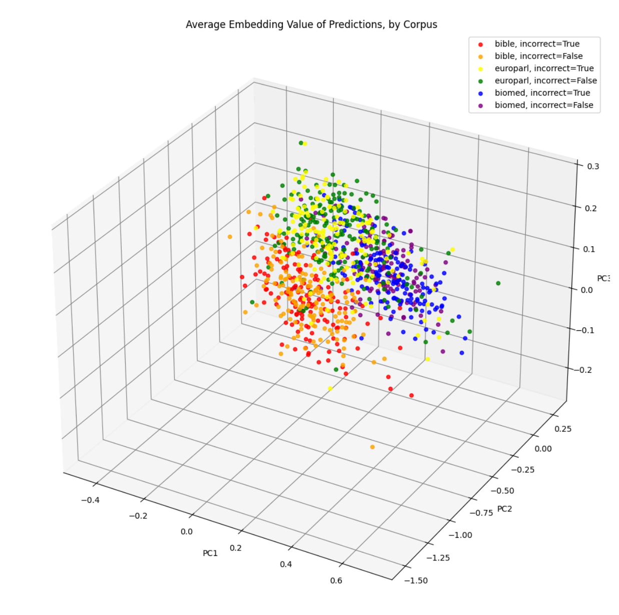
*KL Divergence metrics drawn from a single evaluation



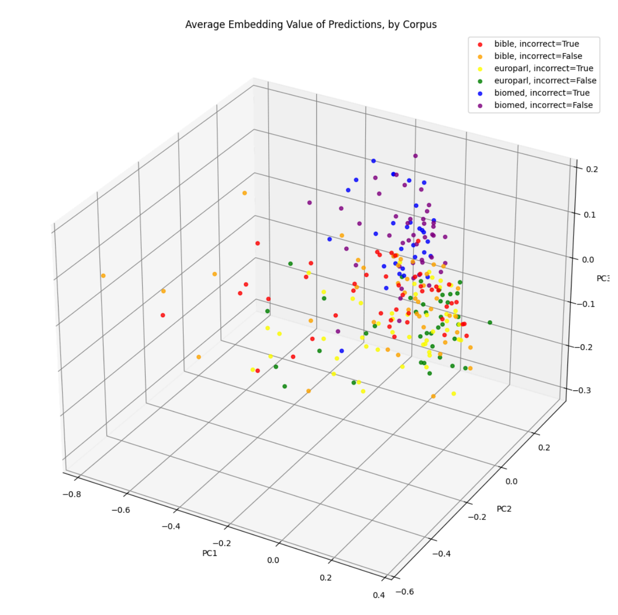
Appendix F: Misclassification of Sentence Representation Errors in Embedding Space

*BERT and ModernBERT's embeddings are on different scales, and KL Divergence metrics drawn from a single evaluation

Single Task	Multi Task
BERT-base-cased	BERT-base-cased
 'biomed' symmetric KL divergence: 3.244608891516798e-07 'europarl' symmetric KL divergence: 5.228060469033662e-07 'bible' symmetric KL divergence: 6.683676775615276e-07	 'bible' symmetric KL divergence: 2.95432516672069e-06 'biomed' symmetric KL divergence: 1.6558011492547724e-06 'europarl' symmetric KL divergence: 3.947562264288915e-06
BERT-large-cased	BERT-large-cased
 'biomed' symmetric KL divergence: 2.593201681598221e-07 'europarl' symmetric KL divergence: 4.6426749472826705e-07 'bible' symmetric KL divergence: 6.619976385796855e-07	 'bible' symmetric KL divergence: 1.888256876176576e-06 'biomed' symmetric KL divergence: 1.6649774807960557e-06 'europarl' symmetric KL divergence: 3.837849675757952e-06
ModernBERT-base	ModernBERT-base



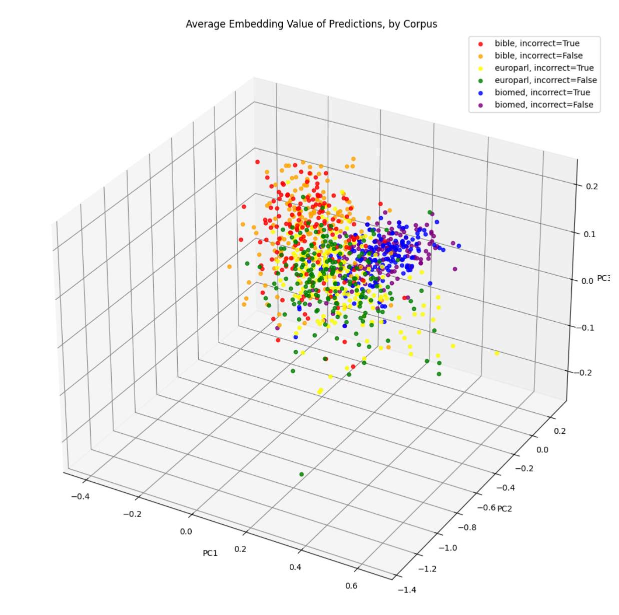
'biomed' symmetric KL divergence: 1.8457217644529404e-06
'europarl' symmetric KL divergence: 1.4357515857590244e-06
'bible' symmetric KL divergence: 1.6860511154101313e-06



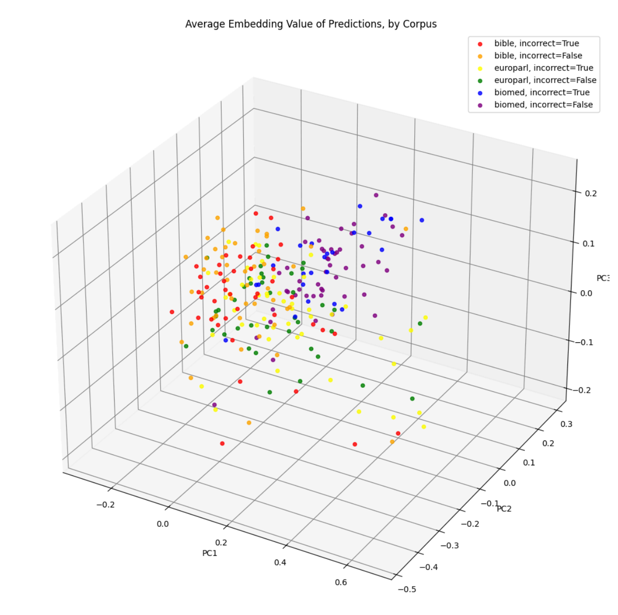
'bible' symmetric KL divergence: 8.12416918642706e-06
'biomed' symmetric KL divergence: 6.612745022528127e-06
'europarl' symmetric KL divergence: 1.2013919929424917e-05

ModernBERT-large

ModernBERT-large



'biomed' symmetric KL divergence: 8.158935155510512e-07
'europarl' symmetric KL divergence: 9.710584787971171e-07
'bible' symmetric KL divergence: 1.1354024662615837e-06



'bible' symmetric KL divergence: 4.29344419874927e-06
'biomed' symmetric KL divergence: 3.5779938949494955e-06
'europarl' symmetric KL divergence: 4.058646832767972e-06