Groningen Group E at SemEval-2024 Task 8: Detecting machine-generated texts through pre-trained language models augmented with explicit linguistic-stylistic features

Patrick Darwinkel, Sijbren van Vaals, Marieke van der Holt, Jarno van Houten
University of Groningen
Broerstraat 5, 9712 CP Groningen
{p.darwinkel, s.j.van.vaals, m.van.der.holt, j.t.van.houten}@student.rug.nl

Abstract

Our approach to detecting machine-generated text for the SemEval-2024 Task 8 combines a wide range of linguistic-stylistic features with pre-trained language models (PLM). Experiments using random forests and PLMs resulted in an augmented DistilBERT system for subtask A and B and an augmented Longformer for subtask C. These systems achieved accuracies of 0.63 and 0.77 for the mono- and multilingual tracks of subtask A, 0.64 for subtask B and a MAE of 26.07 for subtask C. Although lower than the task organizer's baselines, we demonstrate that linguistic-stylistic features are predictors for whether a text was authored by a model (and if so, which one).

1 Introduction

The SemEval-2024 Task 8 is aimed at the detection of machine-generated texts across different domains, languages, and generators. The challenge of distinguishing machine-generated from human written texts has become increasingly relevant with the rapid improvement and coinciding widespread usage of Large Language Models (LLMs) such as ChatGPT. Detection of machine-generated text can be important to uncover the purposeful spreading of misinformation on social media, or fraudulent articles and papers in the context of journalism and academics (Tang et al., 2023). To this end, the task organizers collect human data from Wikipedia, Reddit, Wikihow, PeerRead, and ArXiv abstracts in English, Chinese, Urdu, Russian, Indonesian, Arabic, and Bulgarian (Wang et al., 2023b). Consequently, Wang et al. (2023b) prompted 5 different generative LLMs to write the corresponding posts or abstracts based on the titles. The shared task consists of 3 subtasks. Subtask A is the task of classifying between human and machine-generated texts, subtask B pertains pointing out which LLM (or human) generated the text specifically, and lastly, subtask C is about determining the boundary where a text switches from human written to machine-generated.

The focus for our submission to the shared task is to investigate and compare the linguistic-stylistic characteristics of various LLMs, given that previous literature has shown that text produced by generative LLMs contain linguistic-stylistic anomalies (Tang et al., 2023). Additionally, we explore ways to combine features with the power of a pre-trained language model (PLM). Although a system inspired by linguistic-stylistic features may not achieve the greatest scores, it may perform well across domains and is highly interpretable. Additionally, the performance with linguistic-stylistic features may differ per LLM and per domain, for which they possibly yield interesting insights and contribute to scientific knowledge regarding what LLM-generated anomalies consist of.

Ultimately, our system yields passable results, coming in at 110 and 41 for subtask A mono- and multilingual respectively, and ranking at 46 for subtask B and 20 for subtask C. Contrary to expectations, the system appears to be relatively poor at generalizing between domains, but markedly better at dealing with multiple languages, as indicated by the increase in accuracy as well as ranking between the mono- and multilingual conditions in subtask A.

2 Background

To investigate which features contribute to the detection of machine-generated text, we collected 20 metrics from previous research which seem relevant. These features can be broadly divided into 6 categories, which will each be presented in this section.

2.1 Readability

Studies have shown that LLMs are capable of producing more readable text than human professionals when it comes to complex matters such as informed consent documentation (Decker et al., 2023). Pu and Demberg (2023) have also shown that, when it comes to producing summaries for layman or experts, ChatGPT tends to score very similar in both conditions, whereas human summarizers achieve lower scores for laymen, and higher in the expert condition. For this reason, we have selected 3 common, yet distinct readability formulas. The Flesch-Kincaid score for reading ease by taking the average sentence length, and the average number of sylables per word (Kincaid et al., 1975). Similarly, the Coleman-Liau index takes into account the average sentence and word lengths to compute a score (Coleman and Liau, 1975). Lastly, the Dale-Chall Readability Score is calculated using the average sentence length and the ratio of difficult words from a list to the total number of words (Chall and Dale, 1996).

2.2 Entity recognition

While previous research on using LLMs for Named Entity Recognition has shown promising results (Wang et al., 2023a), LLMs are still not as good as humans annotators. Therefore, we expect there may be a difference between human and machine-generated texts when it comes to entities. For this reason, we incorporate the ratio of entities to total words, as well as the ratio of unique to total number of entities as features.

2.3 Syntax

Syntax is concerned with the way words are put together to form proper sentences, often operationalized through dependency parsing where sentence constituents are labelled and linked to determine the syntactic structure of a sentence. Pu and Demberg (2023) used ChatGPT to transform texts from formal to informal and vice versa and found a clear difference between ChatGPT-generated and human-written text in the dependencies for both formal and informal sentences. Therefore, we include several metrics using dependency parsing. Firstly, we consider the average parse tree height (the length of the longest series of dependencies from the root constituent of a sentence). Additionally, we include the average number of noun phrases per sentence. Lastly we employ a measure of syntactic complexity, namely the Coh-Metrix SYNNP index (Graesser et al., 2004), which measures the mean number of modifiers per noun-phrase to compute complexity.

2.4 Semantics

Aside from syntactic features, previous research has also indicated differences on a semantic level. Firstly, machine-generated texts are less coherent than their human written counterparts (Tang et al., 2023). To make the concept of coherence measurable, we adopt the notion of lexical chains (Morris and Hirst, 1991), which refers to a series of related words that are linked by a common thread of meaning. The relevant features based on lexical chains are the total number, the average length and the span of lexical chains in a document. Furthermore, research has shown that ChatGPT produces less negative sentiment and offensive speech compared to human-authored texts (Tang et al., 2023), so we also include a score for controversy as a feature.

2.5 Text length statistics

As an extension of the readability metrics (see Section 2.1), which mostly combine different statistical features of texts to compute a score, we also take into account individual statistics of the document. Specifically, the average number of syllables per word, and the average sentence length.

2.6 Lexical Richness

LLMs work by selecting high-likelihood words to create coherent texts, making it likely for them to write using a lower diversity of words than humans. Previous research has shown that this is indeed the case (Guo et al., 2023). For this reason, we include a number of measures of lexical richness. Firstly, the type token ratio (TTR) and secondly, as an alternative to the TTR, which is sensitive to a steep drop-off in longer texts, the Measure of Lexical Diversity in Text (MLTD). To further study the lexical diversity put forth by LLMs, we consider the hapax richness of the document, which is the ratio of words in the text that occur only once. Lastly, we examine the ratio of function words to content words.

3 System overview

The backbone of our system is combining a feature-driven approach with the state-of-the-art in text classification, namely PLMs. To accomplish this, we tested three main system architectures for this shared task. Firstly, we augment DistilBERT with the feature set. Similarly, to achieve token-level classification, we perform the same for Longformer, and lastly, we use a Random Forest classifier with

DistilBERT embeddings in conjunction with the feature set. For comparison the features were also used separately and in combination with unigrams in a random forest classifier. The following sections will go into detail on each of these individually.

3.1 Augmented DistilBERT

For subtasks A and B, we augment `distilbert-base-cased` (Sanh et al., 2020) for the monolingual tasks and `distilbert-base-multilingual-cased` for the multilingual task with an additional layer for classification using features. The 20 features are run through a linear layer with ReLU activation. The output from this linear layer, which is equal in dimensions to DistilBERT's configured hidden size, is concatenated to the pooled output from DistilBERT's final hidden state. This results in a new tensor of 2*hidden size. This tensor is fed into another linear layer (2*hidden size, hidden size) with ReLU activation and dropout. The output from this is fed into the final classifier layer (hidden size, amount of labels).

3.2 Augmented Longformer

A challenge we ran into is the application of sentence-based features to token-level classification in subtask C. To address this problem, we use an augmented version of Longformer (Beltagy et al., 2020). The architecture of the Longformer is similar to the augmented DistilBERT, except that the output from the extra features is concatenated to the output state of each of the tokens separately. This essentially augments each token with contextual knowledge of the linguistic characteristics of the text that they occur in, enabling token-level classification.

3.3 DistilBERT-embedded Random Forest

Next to the augmented DistilBERT for subtasks A and B, we explore the use of a Random Forest classifier using `distilbert-base-cased` embeddings, instead of simpler one-hot encodings or TF-IDF embeddings, contatenated with our 20 linguistic-stylistic features. After tokenizing each text, we extract the DistilBERT embeddings for the first 512 sub-word tokens and average them using a concatenation of mean, max, sum and L2 (Euclidean norm) pooling. After retrieving the embeddings, we concatenate them with the feature vector composed of our 20 linguistic-stylistic features. We then use the concatenated embeddings with the features to fit a Random Forest classifier.

We experimented with different configurations which differed in the use of the layer (or hidden state) and pooling technique. We found the first hidden state layer (i.e., the layer after the input layer) using a concatenation of mean, max, sum and L2 pooling to produce the most satisfactory results. Similar to the augmented DistilBERT, we use `distilbert-base-multilingual-cased` in the multilingual track of subtask A.

4 Experimental setup

Much of the experimental setup is similar to the format dictated by shared task organizers (Wang et al., 2024). In particular, the provided train and dev sets were used as-is. However, some relevant aspects for our specific system will be presented in this section.

First and foremost, the features are extracted using a variety of external libraries, including SpaCy and fasttext, and subsequently put into JSON format. Furthermore, in the multilingual track, we recognize the language in question using Stanza and fasttext (how and why these models were used is explained in Appendix B.1), and apply language-specific feature extraction methods accordingly. Features that only work for English (e.g., Dale Chall) are not included in the multilingual track. These features were all assigned a value of -1. An overview of the features and how they were calculated can be found in Appendix B. To compare the features, we measured importance using Mean Decrease in Impurity (MDI), which computes the average change in homogeneity in Random Forest nodes for each feature. The systems will be evaluated using the official metrics of the task: accuracy for subtasks A an B and mean absolute error for subtask C.

5 Results

5.1 Development results

The results of our systems for subtasks A and B can be found in Table 1. For subtask A monolingual, augmented DistilBERT works best with an accuracy of 0.75, slightly better than the baseline of 0.74 from the organizers (Wang et al., 2024). Curiously, it performs worse than the default DistilBERT in the multilingual task. Where the default system had an accuracy of 0.71, the augmented version only had an accuracy of 0.67. Possible explanations are that not all features could be used multilingually

	Random Forest				DistilBERT	
	uni.	feat.	uni. + feat.	emb. + feat.	base	augm.
Subtask A monolingual						
Human	0.70	0.65	0.67	0.46		0.79
Machine	0.49	0.33	0.32	0.71		0.69
Accuracy	0.62	0.54	0.56	0.62	0.69	**0.75**
Subtask A multilingual						
Human	0.62	0.45	0.57	0.33		0.63
Machine	0.35	0.48	0.51	0.60		0.71
Accuracy	0.52	0.47	0.54	0.50	**0.71**	0.67
Subtask B						
Human	0.44	0.40	0.47	0.39		0.66
ChatGPT	0.74	0.57	0.73	0.59		0.77
Cohere	0.40	0.38	0.35	0.31		0.50
Davinci	0.58	0.11	0.40	0.24		0.29
Bloomz	0.93	0.89	0.89	0.83		0.95
Dolly	0.44	0.44	0.51	0.54		0.63
Accuracy	0.60	0.49	0.58	0.51	0.60	**0.64**

Table 1: F1-scores and accuracy for subtask A (monolingual and multilingual) and subtask B on the development sets for all our experiments: random forest with unigrams, features, unigrams + features and embeddings + features, and a base and augmented distilBERT model.

or that the quality of the multilingual features is worse than for the monolingual data, since not every language has the same quality parsing models available.

From the different implementations of the random forest model, the best accuracy on the monolingual task was achieved by both the unigram model and the model using DistilBERT embeddings and features with an accuracy of 0.62. There is a sizeable difference between the f1-scores of the 'human' and 'machine' labels. For the models using either unigrams, features or both, the 'human' label has an f1-score between 0.67 and 0.70, where the 'machine' label only scores between 0.32 and 0.49. Interestingly, adding the DistilBERT embeddings resulted in reversed scores. For this model the 'human' label only obtained an f1-score of 0.46 and the 'machine' label 0.71.

For the multilingual track the best random forest model was the one using unigrams and features which had an accuracy of 0.54. The model using only features performed worst with an accuracy of only 0.47. The reversal of the f1-scores of 'human' and 'machine' labels that occurred one the monolingual data is also present here but only when comparing the unigram model with the model using embeddings and features. The model using only features and unigrams and features both have similar scores for both labels.

Augmented DistilBERT also worked best for subtask B with an accuracy of 0.64. Of the random forest models, there was no model using features that outperformed the model using only unigrams. Of the different sources, davinci was the most difficult to predict with an f1-score of only 0.29 from augmented DistilBERT. The fact that the provided development set almost exclusively contains examples of Bloomz is clearly visible based on the comparatively high f1 scores for that class ranging between 0.83 and 0.95.

Since our features are document-based and subtask C is a token-level classification task, we only have have the results from our baseline Longformer and the augmented Longformer. The augmented system had a mean absolute error (MAE) of 5.29 on the development set. Much better than our non-augmented baseline system which had a MAE of 16.62, but still worse than the organiser's baseline of 3.53 (Wang et al., 2024).

5.1.1 Feature importance

Figure 1 shows the feature importances for subtasks A and B. Similar trends can be seen for both tasks, although there are some differences as well. Type-token ratio (and its extension, MTLD) and the amount and length of lexical chains are important for all tasks. The average number of syllables per word is very important for the monolingual track of subtask A, but not so much for subtask B. Unfortunately this feature could not be used for the multilingual track so a comparison is not possible. For further analysis, a correlation heatmap can be found in Appendix D.

5.2 Test results

The predictions of augmented DistilBERT were sent in for this task for both the monolingual and the multilingual track. For the monolingual track this resulted in an accuracy of 0.63, placing us at position 110 on the leaderboard. For the multilingual track of subtask A, the system performed better with an accuracy of 0.77, leading to position 41. Unfortunately both scores are lower than the organisers' baseline of 0.88 and 0.81 respectively.

For subtask B the same system was used as for subtask A: augmented DistilBERT. The accuracy on the test set was the same as on the development set, namely 0.64. This is lower than the organisers' baseline of 0.75 unfortunately. For this task it is interesting to look at a confusion matrix which is shown in Figure 2. It shows that there were a few

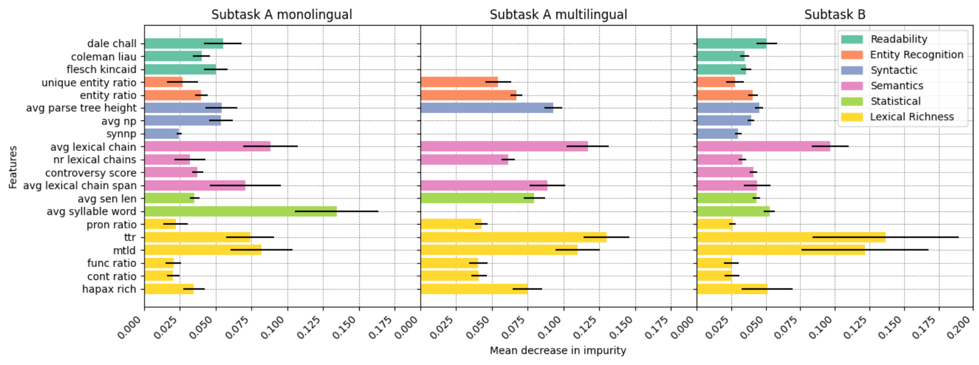


Figure 1: Feature importance (mean decrease in impurity from a random forest model using only our features) for subtasks A and B on the train set. The features are grouped by category.

generators that were problematic for the system. Cohere was almost never classified correctly and was mostly classified as ChatGPT. Dolly was often misclassified as davinci. Human written text was often classified as generated by either davinci or dolly, but almost never as the other three generators. For this subtask we obtained position 46 on the leaderboard.

We also evaluated the performance of our other main system for subtasks A and B, the random forest with embeddings and features, on the test set. This resulted in accuracies of 0.59 and 0.65 for subtask A (mono- and multilingual) and 0.39 for subtask B, also featured in Table 2. A possible explanation for the lower accuracies is that only the first 512 sub-word (BPE) tokens were taken into account. A lot of information gets lost during pooling and combing different strategies could not prevent a severe loss of information.

For subtask C the augmented Longformer achieved a MAE of 26.07, ranking us at position 20. Unfortunately this system also did not outperform the organiser's baseline of 21.54.

Subtask	Baseline	Augm. DistilBERT	Emb. RF
A mono	0.88	0.63	0.59
A multi	0.81	0.77	0.65
B	0.75	0.64	0.39

Table 2: Accuracy of the baseline, augmented DistilBERT and the embedded random forest on the test set for subtasks A and B.

6 Conclusion

Unfortunately none of our systems performed better than the baseline but our experiments did give some insight in how features can be used. Our final systems producing the most satisfactory results for

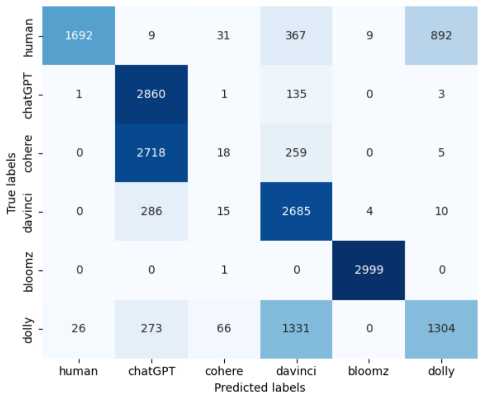


Figure 2: Confusion matrix from augmented DistilBERT for subtask B on the test set.

the test set were: the augmented DistilBERT system for subtasks A and B, resulting in accuracies of 0.63 and 0.77 respectively, and the augmented Longformer for subtask C, obtaining a MAE of 26.07.

Contrary to the literature, it does not appear that the feature-based methods are better at generalizing. During our experiments we saw that when the unseen test data is from the same distribution (held from the training data), the feature-based approach performs far better in terms of accuracy (A mono: 0.88; A multi: 0.79; B: 0.71; for a full overview, see Appendix E). This is a clear indication that our proposed stylistic-linguistic features contain sufficient predictive power to distinguish human- from machine-written text. In particular, we find Type-token ratio, MTLD, the amount and length of lexical chains, and the average number of syllables per word to be noteworthy features for the given task. Future research into (different) features, especially for multilingual tasks could therefore be fruitful.

References

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer.

J. S. Chall and E. Dale. 1996. Readability revisited : The new dale-chall readability formula., brookline books.

M. Coleman and T. L. Liau. 1975. A computer readability formula designed for machine scoring.

Hannah Decker, Karen Trang, Joel Ramirez, Alexis Colley, Logan Pierce, Melissa Coleman, Tasce Bongiovanni, Genevieve B. Melton, and Elizabeth Wick. 2023. Large Language ModelBased Chatbot vs Surgeon-Generated Informed Consent Documentation for Common Procedures. *JAMA Network Open*, 6(10):e2336997–e2336997.

Arthur C. Graesser, Danielle S. McNamara, Max M. Louwerse, and Zhiqiang Cai. 2004. Coh-metrix: Analysis of text on cohesion and language. *Behavior Research Methods, Instruments, & Computers*, 36(2):193–202.

Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. 2023. How close is chatgpt to human experts? comparison corpus, evaluation, and detection.

Peter Kincaid, Robert P. Fishburne, Richard L. Rogers, and Brad S. Chissom. 1975. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel.

Jane Morris and Graeme Hirst. 1991. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics*, 17(1):21–48.

Dongqi Pu and Vera Demberg. 2023. ChatGPT vs human-authored text: Insights into controllable text summarization and sentence style transfer. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pages 1–18, Toronto, Canada. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

Ruixiang Tang, Yu-Neng Chuang, and Xia Hu. 2023. The science of detecting llm-generated texts.

Shuhe Wang, Xiaofei Sun, Xiaoya Li, Rongbin Ouyang, Fei Wu, Tianwei Zhang, Jiwei Li, and Guoyin Wang. 2023a. Gpt-ner: Named entity recognition via large language models.

Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Osama Mohammed Afzal, Tarek Mahmoud, Giovanni Puccetti, Thomas Arnold, Chenxi Whitehouse, Alham Fikri Aji, Nizar Habash, Iryna Gurevych, and Preslav Nakov. 2024. Semeval-2024 task 8: Multigenerator, multidomain, and multilingual black-box machine-generated text detection. In *Proceedings of the 18th International Workshop on Semantic Evaluation*, SemEval 2024, Mexico, Mexico.

Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Chenxi Whitehouse, Osama Mohammed Afzal, Tarek Mahmoud, Alham Fikri Aji, and Preslav Nakov. 2023b. M4: Multi-generator, multi-domain, and multi-lingual black-box machine-generated text detection.

A Acknowledgements

This submission has been carried out as part of the 2023-2024 edition of the master course Shared Task Information Science (LIX026M05) at the University of Groningen, taught by Lukas Edman and Antonio Toral. We thank the Center for Information Technology of the University of Groningen for their support and for providing access to the Hábrók high performance computing cluster.

B Feature Extraction

In this section we describe how our features were calculated, what language models were used and how these models were selected. Version numbers of the specific libraries that were used can be found in appendix C.

B.1 Model selection

For the monolingual subtasks, we use the English `spacy_udpipe` model. This model is used to obtain POS-tags, noun chunks, syllable counts and parse trees. `en_core_news_sm`, also a spacy model, is used for named entity recognition.

For the multilingual task, the documents are first tagged with a language label by stanza's language identification model. This language label was then used to (try to) download the correct `spacy_udpipe` model and when this was not available, the correct stanza model. We did not use stanza models for all languages because they are quite large and slow.

For the NER tagger, the script first tries to download either `language_core_web_sm` or `language_core_news_sm`. When neither of these models is available, a universal model is used, namely `xx_ent_wiki_sm`.

For the fasttext embeddings the language of each document was first detected with ftlangdetect's `detect`. Because the spacy, stanza and fasttext models together take up quite some space, the data was processed per language. For this we chose the stanza-identified language. For each language as it was identified by stanza, we chose the most occurring fasttext-detected language to download the correct fasttext model. It was not possible to use the stanza detected language for this as the language codes were not always identical and the languages supported by both are not the same.

B.2 Readability features

Flesch Kincaid

Note: this feature only works for English.
Calculated using textstat's `flesch_reading_ease()`. The score is normalized by dividing the score by 100 (the maximum score) and subtracting this from 1 to invert the scale.

Coleman Liau

Note: this feature only works for English.
Calculated using textstat's `coleman_liau_index()`. The score is normalized by dividing it by 30 (the maximum score).

Dale Chall

Note: this feature only works for English.
Calculated using textstat's `dale_chall_readability_score()`. The score is normalized by dividing it by 20 (the maximum score).

B.3 Entity Recognition features

Entity Ratio

The number of entities divided by the number of words in a document.

Unique entity ratio

The number of unique entities divided by the total number of entities in a document.

B.4 Syntactic features

Average parse tree height

Average length of the longest series of dependencies from the root constituent of all sentences in the text.

Average number of noun phrases

Average number of noun chunks in a document.

SYNNP

Average number of tokens in a noun chunk in a document.

B.5 Semantic features

Lexical Chains

To create lexical chains only nouns were used. A chain is created by putting words together whose fasttext embedding have a cosine similarity (`sklearn`'s `cosine_similarity`) of more than 0.5. This feature is not used on its own, but to calculate the next three features.

Number of lexical chains

The sum of all lexical chains in a document. The score is normalized by dividing it by the number of words in a document.

Average lexical chain length

The average number of words in a lexical chain in the document. The score is normalized by dividing it by the number of words in a document.

Average lexical chain span length

The span is the number of words in the document between the first and last word of a lexical chain. Of this we take the average. The score is normalized by dividing it by the number of words in a document.

Controversy score

Note: this feature only works for English.
Calculated using `polarity_scores()` from `nltk`'s `SentimentIntensityAnalyzer()`.

B.6 Statistical features

Average number of syllables

Note: this feature only works for English.
Average number of syllables in a token.

Average sentence length

Average number of tokens per sentence (split by a period) in a document.

B.7 Lexical richness features

TTR

Calculated using `LexicalRichness().ttr` from `lexicalrichness`.

MTLD

Calculated using `LexicalRichness().mtld()` from `lexicalrichness`.

Hapax Richness

Calculated by getting `hapaxes()` from `nltk`'s `FreqDist()` and dividing it by the number of tokens in the document.

Content word ratio

The following POS-tags were used to select content words: "NOUN", "PROPN", "VERB", "ADJ", "ADV" and "NUM" from the universal POS tags and "CD", "JJ", JJR", "JJS", "POS", "PRP$", "RB", "RBR", "RBS", "WP$" and "WRB" from the Penn Treebank POS tags. The ratio is calculated by dividing the number of content words by the total number of words.

Function word ratio

Function words are all words that are not content words. The ratio is calculated by dividing the number of function words by the total number of words.

Pronoun ratio

Pronous are selected using the universal POS-tag "PRON" and the Penn Treebank POS-tags "PRP", "PRP$", "WP" and "WP$". The ratio is calculated by dividing the number of pronouns by the total number of words.

C Dependencies

The dependencies with version numbers that were used for both the feature extraction and the different system implementations.

- `fasttext`[1]
- `fasttext-langdetect`==1.0.5
- `lexicalrichness`==0.5.1
- `nltk`==3.8.1
- `numpy`==1.26.3
- `pandas`==2.1.4
- `scikit-learn`==1.3.2
- `spacy-udpipe`==1.0.0
- `spacy_syllables`==3.0.2
- `spacy_stanza`==1.0.4
- `stanza`==1.6.1
- `textstat`==0.7.3
- `datasets`==2.16.1
- `transformers`==4.36.2
- `accelerate`==0.25.0
- `evaluate`==0.4.1

[1]Recent Python and C++ versions require fasttext git:
fasttext @ git+https://github.com/facebookresearch/fastText@6c2204ba66776b700095ff73e3e599a908ffd9c3

Subtask A monolingual	
Human	0.89
Machine	0.87
Accuracy	0.88
Subtask A multilingual	
Human	0.78
Machine	0.80
Accuracy	0.79
Subtask B	
Human	0.76
ChatGPT	0.68
Cohere	0.68
Davinci	0.63
Bloomz	0.94
Dolly	0.54
Accuracy	0.71

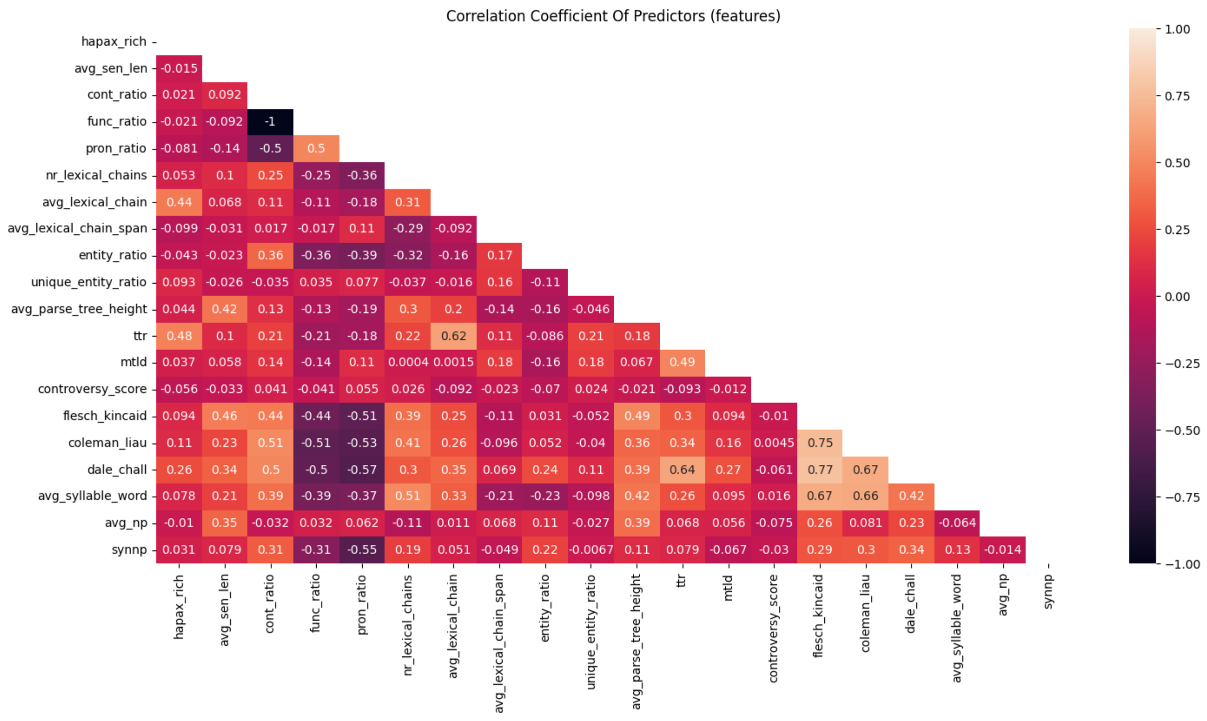
Table 3: F1-scores and accuracy for subtask A (monolingual and multilingual) and subtask B from the random forest model using only features tested on a held out set of 20% of the training data.

D Feature multicolinearity

Figure 3 shows the multicolinearity of the features based on the training data set for the monolingual track of subtask A.

E Feature-based RF results on training data

Table 3 shows the results for subtasks A (monolingual and multilingual) and B when a random selection of 20% of the training set is held out and used as a test set for the random forest model using only features.



	hapax_rich	avg_sen_len	cont_ratio	func_ratio	pron_ratio	nr_lexical_chains	avg_lexical_chain	avg_lexical_chain_span	entity_ratio	unique_entity_ratio	avg_parse_tree_height	ttr	mtld	controversy_score	flesch_kincaid	coleman_liau	dale_chall	avg_syllable_word	avg_np	synnp
hapax_rich																				
avg_sen_len	-0.015																			
cont_ratio	0.021	0.092																		
func_ratio	-0.021	-0.092	-1																	
pron_ratio	-0.081	-0.14	-0.5	0.5																
nr_lexical_chains	0.053	0.1	0.25	-0.25	-0.36															
avg_lexical_chain	0.44	0.068	0.11	-0.11	-0.18	0.31														
avg_lexical_chain_span	-0.099	-0.031	0.017	-0.017	0.11	-0.29	-0.092													
entity_ratio	-0.043	-0.023	0.36	-0.36	-0.39	-0.32	-0.16	0.17												
unique_entity_ratio	0.093	-0.026	-0.035	0.035	0.077	-0.037	-0.016	0.16	-0.11											
avg_parse_tree_height	0.044	0.42	0.13	-0.13	-0.19	0.3	0.2	-0.14	-0.16	-0.046										
ttr	0.48	0.1	0.21	-0.21	-0.18	0.22	0.62	0.11	-0.086	0.21	0.18									
mtld	0.037	0.058	0.14	-0.14	0.11	0.0004	0.0015	0.18	-0.16	0.18	0.067	0.49								
controversy_score	-0.056	-0.033	0.041	-0.041	0.055	0.026	-0.092	-0.023	-0.07	0.024	-0.021	-0.093	-0.012							
flesch_kincaid	0.094	0.46	0.44	-0.44	-0.51	0.39	0.25	-0.11	0.031	-0.052	0.49	0.3	0.094	-0.01						
coleman_liau	0.11	0.23	0.51	-0.51	-0.53	0.41	0.26	-0.096	0.052	-0.04	0.36	0.34	0.16	0.0045	0.75					
dale_chall	0.26	0.34	0.5	-0.5	-0.57	0.3	0.35	0.069	0.24	0.11	0.39	0.64	0.27	-0.061	0.77	0.67				
avg_syllable_word	0.078	0.21	0.39	-0.39	-0.37	0.51	0.33	-0.21	-0.23	-0.098	0.42	0.26	0.095	0.016	0.67	0.66	0.42			
avg_np	-0.01	0.35	-0.032	0.032	0.062	-0.11	0.011	0.068	0.11	-0.027	0.39	0.068	0.056	-0.075	0.26	0.081	0.23	-0.064		
synnp	0.031	0.079	0.31	-0.31	-0.55	0.19	0.051	-0.049	0.22	-0.0067	0.11	0.079	-0.067	-0.03	0.29	0.3	0.34	0.13	-0.014	

Figure 3: Heatmap showing the multicolinearity of the features based on the training data set of subtask A monolingual.