HHUflauschig at GermEval 2025 Shared Task on Candy Speech Detection: Hybrid Approaches for Binary Classification and Span Typing

Wiebke Petersen, Lara Eulenpesch
Heinrich Heine Universität
Düsseldorf, Germany
{wiebke.petersen, lara.eulenpesch}@hhu.de

Abstract

We present our submission to the GermEval 2025 Shared Task on *Candy Speech Detection*, which focuses on identifying and analyzing affectionate language ('Flausch') in German social media comments. For Subtask 1 (binary classification of comments), our approach combines linguistically motivated features with large language models (LLMs). For Subtask 2 (fine-grained span-level detection and categorization), we employ a stacked architecture integrating specialized models for span identification and type classification. Our system achieves second place in Subtask 1 (out of 20 submitted models) and third in Subtask 2 (out of 16 models).

1 Introduction

While the detection of harmful language such as hate speech has received considerable attention in recent years, its positive counterpart remains largely unexplored. The *Candy Speech Detection Shared Task* by Clausen et al. (2025) addresses this gap by introducing the concept of *Flausch* (candy speech), i.e. positive, supportive expressions in online discourse, particularly in German-language YouTube comments.

The task comprises two subtasks: (1) binary classification of whether a comment contains candy speech, and (2) fine-grained span-level detection and categorization of candy speech expressions into one of ten predefined classes.

For Subtask 1, we combined finetuned BERT-based language models with linguistically motivated features such as sentiment and positive word counts. These were fed into a meta-classifier to leverage both learned representations and interpretable features.

For Subtask 2, we compared two strategies: (1) direct fine-tuning of a BERT token classifier with BIO labels, and (2) a two-step pipeline separating span detection and classification. Span detection was implemented via either a fine-tuned BERT model or a rule-based segmentation using dependency parses.

All models were evaluated on a held-out portion of the training data, with only the top-performing system for each subtask submitted to the competition. Our model ranked 2nd in Subtask 1 (out of 20 submissions) and 3rd in Subtask 2 (16 submissions).

2 Related Work

The detection and classification of harmful or toxic content on social media, particularly hate speech, has been extensively studied across multiple languages, including German (e.g., Struß et al., 2019). A wide range of methods have been applied to this task, from keyword-based and tf-idf approaches to shallow classifiers such as logistic regression, support vector machines (SVM), and multi-layer perceptrons. More recently, deep learning and especially transformer-based models have consistently achieved state-of-the-art results (Alkomah and Ma, 2022; Geetanjali and Kumar, 2025; Yin and Zubiaga, 2021).

The SemEval shared task on toxicity span detection (Pavlopoulos et al., 2021) extended toxicity detection and classification to span-level identification, which is closely related to Subtask 2 in our work. As in hate speech detection on the comment or sentence level, transformer models proved most effective for accurately locating toxic spans.

Compared to hate speech, positive or encouraging language such as the *Flausch* or *candy speech* targeted in this shared task has received considerably less attention in NLP. It has been the focus of shared tasks on *hope speech* for multilingual social media content (Chakravarthi et al., 2022; Kumaresan et al., 2023). In those tasks as well, transformer-based models, in combination with tf-idf, SVM and ensemble methods, achieved the strongest results.

3 Data

The dataset released by the task organizers comprises 37,057 German-language social media comments annotated for Flausch-content. We used 90% of this data (33,351 comments) as our training set and held out 10% (3,706 comments) for internal evaluation across all experiments. Throughout the paper, we refer to this set as the *held-out test data*. The official test set provided during the competition consists of 9,229 comments.

For Subtask 1, each comment is labeled with a binary Flausch-label. In Subtask 2, fine-grained span annotations are provided for Flausch-expressing comments, using ten specific categories (e.g., *positive feedback*, *encouragement*, etc.). The official competition test set differs substantially from the training data. On average, comments in the test set are longer (68.6 vs. 58.3 tokens), contain a higher proportion of comments labeled as Flausch (41.3% vs. 29.1%), and include more annotated spans per comment (0.65 vs. 0.43). In addition, the distribution of span types in the test data diverges noticeably from that in the training set (see Appendix A for details).

4 Subtask 1

4.1 System Overview

Our approach to Subtask 1 combines transformer-based models with linguistic feature extraction and a meta-classification step. The following sections describe the key components of our system.

Data Preprocessing. To improve robustness and feature quality, we applied two preprocessing steps to the original German YouTube comments:
Spelling correction: Since YouTube comments often contain numerous typos, non-standard spelling, and informal language, we used a publicly available BART-based spelling correction model[1] to normalize spelling variants and reduce noise in the input data.
English translation: As many NLP models perform better in English or are only available for English, we translated the comments using the German-English model[2] by Tiedemann and Thottingal (2020) to enable cross-lingual feature extraction.

[1] oliverguhr/spelling-correction-german-base
[2] Helsinki-NLP/opus-mt-de-en

Fine-tuning BERT Models. We fine-tuned several transformer-based models for binary classification of candy speech across different comment variants. For the German data, we used the large German BERT model[3] by Chan et al. (2020) and the base German BERT model[4] by the dbmdz team and trained each on both the original and the spelling-corrected comments. In addition, we fine-tuned the large English RoBERTa model[5] by Liu et al. (2019) on the translated versions of the comments to leverage the potential of high-performing English-language models.

For fine-tuning, 15% from our internal training data (see Section 3) were set aside as validation set. All models were fine-tuned for 3 epochs using the Huggingface `Trainer` framework with a batch size of 16, a learning rate of $2 \cdot 10^{-5}$, and weight decay of 0.01. Model selection was based on the best evaluation F1 score on the validation set, with evaluation and logging performed every 500 training steps. All fine-tuned models are publicly available on Huggingface (see Appendix B.1).

Linguistic Features To complement our transformer-based models, we incorporated a set of linguistically motivated features. These include sentiment polarity and emotion scores, as well as various count-based surface indicators. Polarity scores, ranging from -1.0 (most negative) to 1.0 (most positive), were computed using `TextBlobDE` for the German comments (both original and spelling-corrected) and `TextBlob` for their English translations.[6]

Emotion probabilities for Ekman's six basic emotions (*joy*, *sadness*, *anger*, etc.) plus a *neutral* class were derived from the English translations using the RoBERTa-based model by Hartmann (2022).[7]

As a reference for common surface elements in candy speech, lists of positive words (and some short phrases), emojis, and emoticons were generated using ChatGPT-4o.[8] To derive a list of positive tokens, the word list was tokenized using the uncased base German BERT tokenizer.[9] Tokens occurring in more than 2% of non-Flausch-comments

[3] deepset/gbert-large
[4] dbmdz/bert-base-german-cased
[5] FacebookAI/roberta-large
[6] https://github.com/sloria/textblob
[7] j-hartmann/emotion-english-distilroberta-base
[8] Prompt used: "Gib mir drei sehr ausführliche Listen im csv Format für Wörter, Emoticons und Emojis, die häufig in positiven deutschen YouTube Kommentaren vorkommen. Keine Erklärungen oder ähnliches hinzufügen."
[9] dbmdz/bert-base-german-uncased

were excluded to improve specificity.

These resources were used to compute additional features: counts of positive words, positive tokens, emojis, and emoticons per comment. We also computed the ratio of positive tokens relative to total tokens. Finally, two additional surface-level features were included: the number of fully capitalized words and the number of characters repeated three or more times, both of which can be indicative of emphatic or emotionally expressive language often found in candy speech.

Meta-Classifier Finally, we experimented with several meta-classifiers from the `scikit-learn` library to combine the softmax outputs of the fine-tuned BERT models with the extracted linguistic features. To evaluate different combinations of input features and classifiers (e.g., logistic regression, random forest, SVM), we internally split our held-out test set into a training portion and a separate evaluation portion. Since differences in performance were marginal across classifier types, we opted for logistic regression due to its simplicity and interpretability.

4.2 Results

Table 1 summarizes the results of our fine-tuned transformer models on the held-out test data. The best model, `gbert-large` trained on original (uncorrected) comments, achieved an F1 score of 0.906. Performance trends show that (1) models trained on original input outperform those using corrected text, (2) large models outperform base models, and (3) German models perform better on original comments than the English model on translated ones.

Model	**Input**	**Prec.**	**Recall**	**F1**
`gbert-large`	orig.	0.881	**0.932**	**0.906**
`gbert-large`	corr.	**0.887**	0.906	0.896
`bert-base-german`	orig.	0.885	0.884	0.885
`bert-base-german`	corr.	0.881	0.879	0.880
`roberta-large`	transl.	0.885	0.865	0.875

Table 1: Precision, recall, and F1-score of fine-tuned transformer models on different input variants (original comment, spelling corrected, translated) on held-out test data (Subtask 1).

To enhance performance, we trained a logistic regression meta-classifier combining BERT softmax scores with linguistic features (Table 2). It turns out that combining BERT softmax scores with the additional features yields a clear improvement over using the linguistic features alone, and also outperforms the best individual BERT model. The strongest performance was achieved using only the softmax scores of the best-performing BERT model (`gbert-large`), all sentiment scores (polarity and Ekman's emotions), and positive word/token statistics. This feature configuration was used in our final competition submission.

Features	**Prec.**	**Rec.**	**F1**
all sentiments (Ekman + polar.)	0.754	0.519	0.615
all non-BERT features	0.785	0.621	0.694
all features (incl. BERT)	0.936	0.927	0.932
all BERT features	**0.944**	0.908	0.926
`gbert-large` comment + all non-BERT features	0.920	**0.947**	0.933
`gbert-large` comment + all sentiments + positive word count + positive token count + positive token ratio ($*$)	0.929	**0.947**	**0.938**

Table 2: Performance of logistic regression meta-classifier with different feature combinations on our held-out test data. Submitted model marked with ($*$).

On the official competition test set, the final model achieved a precision of 0.900, recall of 0.875, and F1 score of 0.887 – slightly below our held-out results but still strong. Our system ranked second out of 20 submissions for Subtask 1. The drop in performance likely reflects distributional differences between training and competition data (see Section 3) and a non-ideal selection of held-out test data (see Section 6).

5 Subtask 2

5.1 System Overview

We approach Subtask 2 by a single-step model that performs joint span detection and classification with a token classifier as well as by two-step approaches that decouple span detection and classification.

1-step model (`flausch-span-end-to-end`). This model is based on `gbert-large` and fine-tuned using the BIO tagging scheme with type-specific labels (`B-type`, `I-type`, `O`), where `type` corresponds to one of the ten Flausch-categories. Training and evaluation are performed on the same splits as in Subtask 1, with 15% of the internal training data held out for validation. The model is optimized for the F1-score using standard parameters and an evaluation frequency of 500 steps. Performance scores of this model are given in Table 3.

Dependency-based 2-step model (`spacy 2-step`). Our first two-step model combines rule-based span segmentation based on dependency parsing with a learned classifier for span categorization. In the first step, each comment is parsed using spaCy's dependency parser. Span segmentation is then performed by assigning each token to the root of its syntactic subtree. A token is considered a span root if it is either the sentence's syntactic root or has a dependency relation such as 'reported speech' (rs), 'coordinating conjunction' (cd), or 'junctor' (ju). For each token, we recursively follow its syntactic head until a span root is reached.[10] Tokens sharing the same span root and occurring consecutively in the comment are grouped into a span.

In the second step, the extracted spans are classified using a `gbert-large`-based sequence classifier (`span-classifier-with-none`), trained to assign one of ten fine-grained Flausch categories or the additional `not-flausch` class. The training and validation data are based on the same split used in Subtask 1 and are constructed as follows: (1) For comments annotated as containing Flausch (`flausch: yes`), we label the gold Flausch spans with their respective subtype. The remaining contiguous sequences of tokens between these gold spans are treated as separate spans and labeled `not-flausch`. (2) For comments labeled as not containing Flausch, we apply our spaCy-based span segmentation method and label all resulting spans as `not-flausch`.

The classifier is fine-tuned to optimize macro-F1, using standard hyperparameters and an evaluation interval of 1000 steps. On our held-out test set, the model achieves a weighted F1 score of 0.95, but a substantially lower macro F1 score of 0.66. This discrepancy reflects the extreme label imbalance in the training data (e.g., 92 *sympathy*-spans and 55,501 *not-flausch*-spans).

BERT-based 2-step model (1st variant, `gbert 2-step`). In this approach, Flausch-spans are detected in the first step using a dedicated span detector model (`span-detector`). The model is fine-tuned with BIO labels indicating generic span boundaries: `B-span`, `I-span`, and `O`. The training and validation data are the same as in the single-step model, with labels converted accordingly. The span detector is fine-tuned with the same hyperparameters as the single-step model, except for a longer evaluation interval. It reaches an F1 score of 0.77 on our held-out test data.

In the second step, detected Flausch-spans are classified by a multinomial sequence classifier (`span-classifier`) into one of the ten Flausch-types. This classifier is trained on gold spans from the training data and uses the same hyperparameters as the classifier `span-classifier-with-none` in the previous paragraph, but with an evaluation every 500 steps. On our held-out test data it achieves a weighted average F1 score of 0.92 and a macro F1 score of 0.78.

BERT-based 2-step model (2nd variant, `gbert 2-step + not-flausch`). This model combines the previous two-step variants. Span detection is performed using the trained Flausch-span detector `span-detector`, and classification is done using the more general span classifier `span-classifier-with-none`, which is able to assign the label `not-flausch`. The rationale is that the classifier can correct for Flausch-span detection errors by rejecting non-Flausch-spans.

5.2 Results

For Subtask 2, we evaluated both end-to-end models and two-step pipelines. Table 3 summarizes the global results on our held-out evaluation set using three metrics:[11] *Strict* (exact match of span and type), *Span* (correct span regardless of type), and *Type* (correct type regardless of span boundaries).[12]

System	Strict	Span	Type
`flausch-span-end-to-end`	0.647	0.682	0.792
`gbert 2-step` (∗)	**0.728**	**0.769**	**0.833**
`gbert 2-step + not-flausch`	0.693	**0.769**	0.785
`spacy 2-step`	0.370	0.389	0.733

Table 3: Global evaluation results (Strict, Span, and Type-level F1) for Subtask 2 on held-out data. Submitted marked with (∗).

Our best performing system was a two-step BERT-based pipeline (`gbert 2-step`) with sep-

[10]In pseudo code:

```
function ROOT(token)
    if token.dep in ["ROOT", "rs", "cd", "ju"] then
        return token
    else
        return ROOT(token.head)
```

[11]Appendix C provides detailed per Flausch-type label results.

[12]Detailed results for the individual span detector and classifier models are provided in Appendix B.2.

arate span detection and classification modules. It achieved a strict F1 score of 0.728 on our held-out data, outperforming the end-to-end baseline (`flausch-span-end-to-end`) by 8 points. Submitted to the official competition, this model attained a strict F1 of 0.615, a type F1 of 0.766, and a span F1 of 0.668, ranking third among 16 submissions. The drop in performance can be attributed to considerable differences in the Flausch-type distribution between training and test data (see Section A), as well as to the lack of video-level stratification in the held-out test set (see Section 6).

Introducing an additional `not-flausch` label for rejecting non-Flausch spans (`gbert 2-step + not-flausch`) did not improve span-level results and even degraded type-level accuracy. The unbalanced training data, with an overrepresentation of `not-flausch` examples, reduced the model's ability to assign correct Flausch subtypes, without yielding gains in span detection accuracy.

Finally, a rule-based approach using SpaCy dependency parsing for span segmentation produced comparatively weaker results. While conceptually appealing, we believe this method could be enhanced through a more detailed error analysis and by refining the set of rules used for decomposition of the dependency trees.

6 Limitations

Our approach faces several limitations that affect generalization and evaluation reliability:

First, the class distributions in the official training and test sets differ substantially, particularly in the proportion and type of Flausch-comments (see Appendix A). This mismatch limits comparability between held-out and final test performance and likely contributed to the performance drop on the competition set in both subtasks.

Second, our held-out evaluation set was created by randomly sampling 10% of annotated comments without stratifying by video. As a result, comments from the same YouTube videos may appear in both training and evaluation sets, enabling models to exploit video-specific associations (e.g., linking certain creators such as "Die Lochis" to higher probabilities of Flausch-speech) rather than learning generalizable patterns. This setup introduces a form of data leakage and may lead to overly optimistic performance estimates. A more robust design would reserve entire videos for evaluation.

Third, we did not perform a systematic error analysis across models, which limits our understanding of typical failure modes. This is particularly critical for the rule-based span segmentation using SpaCy dependency parsing. The current rule set is heuristic and lacks empirical validation. A detailed qualitative analysis of typical errors (e.g., oversegmentation, incorrect boundary detection) could inform improvements. While we believe this linguistically motivated method holds promise, it currently lacks empirical tuning and optimization.

7 Conclusion

Our submission[13] demonstrates the value of combining linguistic features with LLMs and of structuring the problem into modular components. For Subtask 1, integrating linguistically motivated features led to clear improvements over LLMs alone. For Subtask 2, a two-step pipeline separating span detection and classification outperformed the end-to-end approach. While rule-based span detection using dependency structures showed limited success, we believe that this line of work holds promise and could benefit substantially from more targeted error analysis and refined heuristics.

References

Fatimah Alkomah and Xiaogang Ma. 2022. A literature review of textual hate speech detection methods and datasets. *Information*, 13(6).

Bharathi Raja Chakravarthi, Vigneshwaran Muralidaran, Ruba Priyadharshini, Subalalitha Cn, John McCrae, Miguel Ángel García, Salud María Jiménez-Zafra, Rafael Valencia-García, Prasanna Kumaresan, Rahul Ponnusamy, Daniel García-Baena, and José García-Díaz. 2022. Overview of the shared task on hope speech detection for equality, diversity, and inclusion. In *Proceedings of the Second Workshop on Language Technology for Equality, Diversity and Inclusion*, pages 378–388, Dublin, Ireland. Association for Computational Linguistics.

Branden Chan, Stefan Schweter, and Timo Möller. 2020. German's next language model. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6788–6796, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yulia Clausen, Tatjana Scheffler, and Michael Wiegand. 2025. Overview of the GermEval 2025 Shared Task on Candy Speech Detection. In *Proceedings of the 21st Conference on Natural Language Processing*

[13]Code available at https://github.com/WiebkePetersen/GermevalFlausch

(KONVENS 2025): Workshops, Hildesheim, Germany. ACL.

Geetanjali and Mohit Kumar. 2025. Exploring hate speech detection: challenges, resources, current research and future directions. *Multimedia Tools and Applications*.

Jochen Hartmann. 2022. Emotion English DistilRoBERTa-base. https://huggingface.co/j-hartmann/emotion-english-distilroberta-base/.

Prasanna Kumar Kumaresan, Bharathi Raja Chakravarthi, Subalalitha Cn, Miguel Ángel García-Cumbreras, Salud María Jiménez Zafra, José Antonio García-Díaz, Rafael Valencia-García, Momchil Hardalov, Ivan Koychev, Preslav Nakov, Daniel García-Baena, and Kishore Kumar Ponnusamy. 2023. Overview of the shared task on hope speech detection for equality, diversity, and inclusion. In *Proceedings of the Third Workshop on Language Technology for Equality, Diversity and Inclusion*, pages 47–53, Varna, Bulgaria. INCOMA Ltd., Shoumen, Bulgaria.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized bert pretraining approach.

John Pavlopoulos, Jeffrey Sorensen, Léo Laugier, and Ion Androutsopoulos. 2021. SemEval-2021 task 5: Toxic spans detection. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 59–69, Online. Association for Computational Linguistics.

Julia Maria Struß, Melanie Siegel, Josef Ruppenhofer, Michael Wiegand, and Manfred Klenner. 2019. Overview of GermEval task 2, 2019 shared task on the identification of offensive language. In German Society for Computational Linguistics, editor, *Proceedings of the 15th Conference on Natural Language Processing (KONVENS) 2019*, pages 354–365. s.a., Nürnberg/Erlangen.

Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT — Building open translation services for the world. In *Proceedings of the 22nd Annual Conferenec of the European Association for Machine Translation (EAMT)*, Lisbon, Portugal.

Wenjie Yin and Arkaitz Zubiaga. 2021. Towards generalisable hate speech detection: a review on obstacles and solutions. *PeerJ Computer Science*, 7:e598.

A Data Statistics

This section provides detailed statistics on the training, held-out test, and competition test sets used in our experiments.

Comment Statistics (Task 1)

Dataset	Flausch Ratio	Avg. Length
Own Train	29.2%	58.3 tokens
Own Test	28.2%	59.0 tokens
Competition Test	41.3%	68.6 tokens

Table 4: Flausch ratios and comment lengths across datasets (Task 1).

Span Statistics (Task 2)

Dataset	Spans per Comment	Avg. Length
Own Train	0.43	28.84 tokens
Own Test	0.41	28.14 tokens
Competition Test	0.65	26.30 tokens

Table 5: Span frequency and average span length per dataset (Task 2).

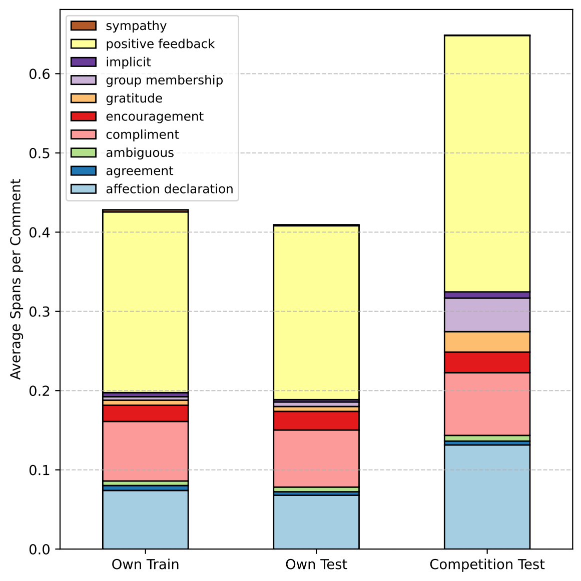

Figure 1: Distribution of fine-grained span types in each dataset. The total height of each bar corresponds to the number of spans per comment.

B Fine-tuned LLMs

B.1 LLMs for Subtask 1

The LLMs finetuned for Subtask 1 are binary classifiers for Flausch-comments. The performances are given in Table 1.

`bert-base-german original` https://huggingface.co/Wiebke/results_flausch_classification_bert-base-german-cased_comment

`bert-base-german spelling corrected` https://huggingface.co/Wiebke/results_flausch_classification_bert-base-german-cased_spelling_corrected

`gbert-large original` https://huggingface.co/Wiebke/results_flausch_classification_gbert-large_comment

`gbert-large spelling corrected` https://huggingface.co/Wiebke/results_flausch_classification_gbert-large_spelling_corrected

`roberta-large translated` https://huggingface.co/Wiebke/results_flausch_classification_roberta-large_translated

B.2 LLMs for Subtask 2

`flausch-span-end-to-end` The `flausch-span-end-to-end` model jointly predicts Flausch-span boundaries and their corresponding types in a single step. It performs reasonably well on all metrics, achieving an F1 of 0.647 (strict), 0.682 (span), and 0.792 (type) on the held-out set. However, performance degrades substantially on the competition test set (strict: 0.544), likely due to domain differences. https://huggingface.co/Wiebke/flausch_span_gbert-large_all

`span-detector` The `span-detector` model identifies Flausch-span locations. It reaches an F1 of 0.769 (span) and 0.768 (strict) on the held-out data, and 0.668 strict F1 on the competition test set. https://huggingface.co/Wiebke/flausch_span_gbert-large_non_labeled_spans

`span-classifier` The `span-classifier` takes gold Flausch-span boundaries and classifies each span into one of the ten fine-grained types. It achieves a macro F1 of 0.78 on the held-out set and 0.71 on the competition test set. Table 6 shows per-class results on both datasets. https://huggingface.co/Wiebke/results_flausch_classification_gbert-large_span_classifier

`span-classifier-with-none` The `span-classifier-with-none` variant includes an explicit `not` class for negative examples and operates on a mixture of annotated and non-annotated spans. While it performs lower overall (macro F1: 0.66), it allows for end-to-end inference when no gold spans are given. Table 7 shows per-class results. https://huggingface.co/Wiebke/task2_flausch_classification_gbert-large_span_classifier_with_nonspan

Label	Held-out F1	Comp. F1
affection declaration	0.91	0.88
agreement	0.94	0.82
ambiguous	0.57	0.61
compliment	0.90	0.83
encouragement	0.92	0.90
gratitude	0.93	0.98
group membership	0.88	0.63
implicit	0.36	0.29
positive feedback	0.95	0.92
sympathy	0.44	0.22

Table 6: Per-label F1 scores for the `span-classifier` model on held-out and competition test sets.

Label	F1 Score	Support
affection declaration	0.85	252
agreement	0.27	16
ambiguous	0.34	22
compliment	0.84	267
encouragement	0.88	87
gratitude	0.75	23
group membership	0.76	21
implicit	0.47	11
not	0.98	6494
positive feedback	0.85	813
sympathy	0.31	5

Table 7: Per-label F1 scores and support counts for the `span-classifier-with-none` model on span data derived from held-out test data.

C Subtask 2: per Flausch-type label results

Label	**flausch-span-end-to-end**	**gbert 2-step**	**gbert 2-step + not**	**spacy 2-step**
positive feedback	0.700	**0.760**	0.756	0.403
affection declaration	0.631	0.734	**0.747**	0.441
group membership	0.421	**0.615**	**0.615**	0.171
encouragement	0.605	0.728	**0.754**	0.239
compliment	0.597	**0.718**	0.710	0.317
ambiguous	0.383	**0.419**	0.387	0.324
implicit	0.125	**0.182**	0.133	0.133
sympathy	0.000	0.143	**0.182**	0.154
agreement	0.091	0.138	**0.174**	0.000
gratitude	0.744	0.800	**0.837**	0.235

Table 8: Strict-level F1 scores per Flausch-type label for Subtask 2 on held-out test data.

Label	**flausch-span-end-to-end**	**gbert 2-step**	**gbert 2-step + not**	**spacy 2-step**
positive feedback	0.877	**0.890**	0.877	0.781
affection declaration	0.843	**0.870**	0.866	0.784
group membership	0.788	**0.800**	0.765	0.581
encouragement	0.875	0.859	**0.893**	0.874
compliment	0.856	**0.877**	0.867	0.800
ambiguous	**0.550**	0.465	0.387	0.324
implicit	0.250	0.364	**0.400**	0.267
sympathy	**0.222**	0.154	0.182	0.154
agreement	**0.182**	0.138	0.174	0.000
gratitude	0.900	0.884	**0.905**	0.720

Table 9: Type-level F1 scores per Flausch-type label for Subtask 2 on held-out test data.