

Data-to-Text: Generating Football Match Reports from Structured Event Data

Mieke Diessner Sören Ducati Valerie Hanfling Luca Scavone Tuna Uğur

Department of Computational Linguistics

Heidelberg University

69120 Heidelberg, Germany

{diessner, ducati, hanfling, scavone, ugr}@cl.uni-heidelberg.de

Abstract

In this work, we investigate whether the IMPECT¹-dataset, containing abstract tracking data of Bundesliga football matches, can be used to generate authentic natural-language football reports (in the style of kicker articles²) through a Data-to-Text pipeline consisting of a Content Selection and a Text Generation part. We manually align the IMPECT-data to existing match reports, defining report-relevant events, and propose a BiLSTM for automatic report content selection. Based on the selection, we use GEMMA to automatically generate match reports and subsequently evaluate the factual consistency of the generated texts using a triplet-based evaluation approach. Our results demonstrate that even though we are able to generate plausible match reports, they fall short in factual consistency and stylistic variability. Our code can be found on GitLab.³

1 Introduction

Large Language Models (LLMs) are increasingly capable of generating coherent natural language texts, with one application being generating text from structured data, known as Data-to-Text. A multitude of different sports generate large amounts of structured data regarding players, results and events, making it a domain that is particularly interesting for Data-to-Text tasks like generating match reports. While sports journalists intuitively select important events for these reports, automatically determining which information should be included remains a challenge.

In this project, we investigate whether it is possible to generate football match reports similar to kicker reports from structured football event data. Following the pipeline architecture for natural language generation by Reiter and Dale (2000), this

project is formulated as a Data-to-Text task divided into two main steps: (1) Content Selection (termed *content determination* in their framework), which determines which information to include in the game reports, and (2) Text Generation (subsuming their *micro-planning* and *surface-realization* stages), where we generate a report based on the selected information using an LLM.

2 Related Work

Data-to-Text Generation is an active research field which has been explored using different approaches in order to automatically generate natural language from data (Sharma et al., 2024). A prominent domain in which Data-to-Text is actively researched is the generation of sports-related texts. Wiseman et al. (2017) introduce the RotoWire corpus, based on NBA box scores paired with professionally written game summaries, which has become a widely used benchmark for sports Data-to-Text Generation (Sharma et al., 2024). Contrastively, instead of aggregated statistics, our input consists of temporally ordered sequences of events, requiring prior selection of relevant information.

Van der Lee et al. (2017) generate Dutch football match reports tailored to specific audiences using a modular, rule- and template based approach, whereas we generate German articles imitating a certain style using an LLM to generate the reports.

3 Data

For all 306 matches of the 2023-24 Bundesliga season, we obtain match reports from kicker² and the respective tracking data from IMPECT¹.

kicker is a German sports publication that features passionate and humorous match reports written by professional sports journalists.

IMPECT provides open-access data on various sports including football, chronicling matches in the form of structured events that feature time,

¹<https://github.com/ImpectAPI/open-data>

²<https://kicker.de>

³<https://gitlab.cl.uni-heidelberg.de/ducati/swp>

action-type, player and more information about every action of a game.

3.1 Dataset Alignment

We create a gold dataset for Content Selection by aligning both datasets, labeling each IMPECT event as either relevant (label 1) or irrelevant (label 0) for the corresponding match-report. This is done in two steps: (1) Text-to-NLGE and (2) NLGE-to-Event(s).

Text-to-NLGE We manually extract Natural Language Game Events (NLGE) from each article. NLGEs are sections of one or more sentences describing a self-contained action that can be linked to a specific time stamp. An example:

Tah köpfte eine Hofmann-Ecke wichtig zum 2:0 ins rechte Eck (35.).^{4 5}

We keep only these NLGEs, leaving us with specific game events that can be aligned with the corresponding structured event data. This leads to an incomplete representation of the games, as we cannot match out-of-game context, atmosphere, etc. to the IMPECT-data.

NLGE-to-Event(s) In the second step, we manually map each NLGE to its corresponding event(s) within the IMPECT data in a one-to-many relationship and tag missing or wrong data. This process reveals severe faults in our data, the exact make-up of which is further explored in Appendix A.2. For our final dataset, we only consider events that are mapped to an NLGE as relevant.

The final dataset is divided into 10 folds for 9-fold cross validation with fold 10 being a held-out test set for all splits. There is a large class imbalance between the two classes: in the training set of split 0, the ratio of negative to positive events is approximately 108:1.

4 Content Selection

We define the selection of report-relevant events as a binary classification task for which we train a classification model on the annotated data. The events that are classified as relevant for a match pose as the input for the text generation task.

⁴EN: "Tah powerfully headed a Hofmann corner into the right corner to make it 2:0 (35.)."

⁵<https://www.kicker.de/leverkusen-gegen-leipzig-2023-bundesliga-4861969/analyse>

4.1 Methods

4.1.1 Manually Defined Rules

As a reference point for evaluating the performance of the subsequent learned models, we investigate to what extent the classification task can be solved by simple, manually defined selection rules, which can be seen in Appendix B.

4.1.2 Random Forest

As a learned baseline, we train a Random Forest classifier (Breiman, 2001), optimizing hyperparameters through grid search in order to avoid baseline nerfing (Leech et al., 2024). The hyperparameter search space is provided in Appendix C.

4.1.3 BiLSTM

Model Architecture We hypothesized that both preceding and subsequent context are important for our classification task (e.g., distinguishing a standard pass from a goal assist) and therefore train a Bidirectional LSTM (BiLSTM; Graves and Schmidhuber, 2005) with a learned attention mechanism for weighted pooling, inspired by Zhou et al. (2016).

Sliding Context Window The input for classifying an event t is a window of surrounding events. Given a window size s , the context window spans events $t - (\lfloor s/2 \rfloor)$ to $t + (\lfloor s/2 \rfloor)$. At the beginning or end of a match, the window is padded to prevent overlap between data from different games.

Training Setup and Hyperparameter Tuning

We train the BiLSTM with the Adam optimizer (Kingma and Ba, 2017) and binary cross-entropy loss with logits (for further training details, see Appendix D). Due to limited computational resources, we conduct a small-scale grid search for six hyperparameters in two rounds. In round 1, we fix two while tuning the other four hyperparameters. Using the best combination from round 1, the two initially fixed parameters are then tuned in round 2. Appendix E provides the exact hyperparameters and search spaces for both rounds.

Downsampling To control for the strong class imbalance (Sicilia et al., 2019), we downsample the majority class with downsampling ratio k during training. For each positive training example, the model only sees k randomly sampled negative ones. The negative training examples are resampled for every epoch.

4.1.4 Feature Selection and Representation

The IMPECT-events each consist of 58 possible features, which are a combination of numeric (e.g. `gameTimeInSec`, `distanceToGoal`), and categorical (e.g. `actionType`, `phase`) values.

Excluding metadata like IDs, we train the Random Forest and the BiLSTM on a full set of 44 features, as well as a feature selection of 22 features that we expect to be relevant to the classification task. For a more detailed breakdown of the manual selection, see Appendix F. Preprocessing details of the numeric and categorical features can be found in Appendix F.1.

Permutation Feature Importance We also analyze feature importances for both the BiLSTM trained on all features and the BiLSTM trained on our feature selection by employing permutation feature importance (Breiman, 2001), for which we permute the values of each feature on the test set for each context window. For this, we randomly sample from all values that the feature takes on in the test set, repeating this setup $K = 10$ times, evaluating only on split 0. We define the permutation feature importance $PFI(f)$ of a feature f as the difference between the model’s F1 score without permutation and the mean of the F1 score with permutation of a feature f on the test set:

$$PFI(f) = F_{1,\text{baseline}} - \frac{1}{K} \sum_{k=1}^K F_{1,\text{permuted}}(f, k) \quad (1)$$

4.1.5 Weighted Tolerance Evaluation Metric

Most relevant events come in sequences that constitute an NLGE. Assuming that deciding whether surrounding events still belong to such a sequence is partially subjective, we define a proximity-aware weighted tolerance metric (Salles et al., 2024; Gim and Min, 2023).

The distance weight $w(d)$ for an event i predicted as positive, i.e., $\hat{y}_i = 1$, at event distance d to the nearest true positive is defined as:

Distance d	0	1	2	3	> 3
Weight $w(d)$	1.0	0.8	0.5	0.1	0.0

The maximum distance of three events is chosen based on the observation that relevant event sequences often span multiple events, e.g. reception–dribble–pass. From these weights, we calculate a weighted tolerance precision, which is combined

with the regular recall to calculate a weighted F1 score.

$$P_w = \frac{1}{N_{\hat{y}=1}} \sum_{i:\hat{y}_i=1} w(d_i) \quad (2)$$

With d_i denoting the event distance from a predicted positive event i to the nearest true positive (relevant) event, $w(d_i)$ is the corresponding tolerance weight and $N_{\hat{y}=1}$ the number of positive predictions.

4.1.6 Aggregated Information

To aid text generation and make sure that certain desired NLGEs will be mentioned in our output, we collect information about each game to accompany the selected events. We aggregate every goal, substitution and red card with the corresponding players and timestamps.

4.2 Results

Table 1 reports F1 score and weighted F1 score (wF1) on the held-out test set, averaged across all splits. The rule-based approach is deterministic and therefore performs identically across splits. Appendix G shows the results for each data split separately.

All four models significantly outperform the rule-based baseline (0.460 F1), and BiLSTMs significantly outperform Random Forests with $p < 0.001^6$. On average, the BiLSTM with the full set of features scores the highest (0.609 F1) with a 14.9pp difference to the baseline.

The Random Forest with full features performs significantly better than the model with selected features on split 0 (0.542 vs. 0.528 F1, $p = 0.03$). The full feature BiLSTM also performs better than the selected model on split 0 (0.614 F1 vs. 0.607 F1), however this result is *not* significant ($p = 0.217$). Given its substantially lower computational cost, we choose the BiLSTM with selected features as our preferred content selection model.

Weighted F1 Score On average, wF1 is between 2.0 and 3.7pp higher than F1 (table 1). As the computation of the weighted F1 score only affects precision and not recall, this gap indicates that a considerable share of each model’s false positives are near-misses rather than fully irrelevant events.

Downsampling Ratio and Window Size Looking at the grid search results of our chosen model,

⁶All significance tests were only conducted on split 0 via paired bootstrap resampling (Koehn, 2004).

Model	Features	F1		wF1	
		Mean	STD	Mean	STD
Rule-Based	–	0.460	0.000	0.480	0.000
RF	selected	0.525	0.004	0.555	0.005
RF	full	0.541	0.003	0.570	0.003
BiLSTM	selected	0.594	0.012	0.630	0.020
BiLSTM	full	0.609	0.011	0.646	0.015

Table 1: Test set results, mean and std over 9 CV splits (fixed test fold). BiLSTM outperforms Random Forest (RF) and the rule-based approach.

the downsampling ratio show no consistent effect on split 0 dev F1 (Figure 4) within the tested range, unlike window size: reducing the window to a single event drops F1 by over 13 percentage points, while window sizes of 11 or more all performed similarly well (Figure 5), confirming that some surrounding context is necessary to perform well on the task but larger windows bring little further benefit.

4.2.1 BiLSTM Error Analysis

Analyzing the BiLSTM’s test set predictions on split 0 along different dimensions, we find that for action types PASS and SHOT, recall for events with a successful outcome (`result = SUCCESS`) is about twice as high as for negative outcomes (see Appendix G.2) while for RECEPTION and SHOT, recall decreases with distance to goal.

A small number of action types rarely have their (few) positively labeled events recognized, while others such as GOAL and OWN_GOAL yield perfect recall and high precision.

Along with our test scores, these findings suggest that the model has learned coarse-grained patterns but sometimes over-relies on surface-level features like `result` and `distance`, struggling with more nuanced cases.

4.2.2 BiLSTM Feature Importances

For both models, `action`, `actionType` and `pxT_team` are among the top 6 features with the highest Permutation Feature Importance (Tables 2 and 3). This also aligns with our initial assumptions about these features when making the manual feature selection (see Appendix F). However, when doing the manual selection, we seemingly underestimated the importance of positional and zone-based features: three of the six most important full-set features (`start_packingZone`, `end_pitchPosition`, `start_lane`) were excluded, and the remaining positional feature `start_pitchPosition` becomes

Feature	Score	
	Mean	STD
<code>actionType</code>	0.0752	0.0041
<code>action</code>	0.0369	0.0047
<code>start_packingZone</code>	0.0197	0.0043
<code>end_pitchPosition</code>	0.0134	0.0037
<code>start_lane</code>	0.0128	0.0056
<code>pxT_team</code>	0.0120	0.0007

Table 2: Top 6 Feature Importance Scores (Full Feature Set).

Feature	Score	
	Mean	STD
<code>start_pitchPosition</code>	0.0348	0.0045
<code>actionType</code>	0.0262	0.0039
<code>action</code>	0.0222	0.0049
<code>pxT_team</code>	0.0201	0.0014
<code>player_position</code>	0.0143	0.0031
<code>distanceToOpponent</code>	0.0080	0.0034

Table 3: Top 6 Feature Importance Scores (Selected Feature Set).

the top feature in the set of manually selected features. For the model trained on the full set of features, importances are distributed among all positional features. This suggests that positional information is meaningful for our classification task.

All feature importances are shown in Appendix F.

5 Text Generation

5.1 Methods

Based on the events classified as relevant during the content selection, the next step in our pipeline is to generate natural language football articles from these using an LLM.

5.1.1 Prompting the Model

The text generation is realized through zero-shot-prompting of GEMMA-4-31B (Gemma-Team, 2026)⁷. The prompt consists of the selected events in their raw JSON format, without any further restructuring or template transformation. Refer to Appendix I for the prompt.

To ensure that GEMMA only relies on the results of the preceding content selection and not on its own inherent knowledge, we test it for parametric knowledge of the Bundesliga season 2023-24 and find it not to be critically contaminated (see Appendix H for all details).

⁷Temperature 1.0, $p = 0.95$, $k = 64$

5.1.2 Text Generation Evaluation

We evaluate the factual consistency of the generated match reports by comparing their individual events to the gold events, representing each event as a triplet containing minute, player ID, and action type information. This approach is based on two previous evaluation methods (Wiseman et al., 2017; Nawrath et al., 2024) and adapted to our football data. Gold triplets are derived from the data, while generated triplets are extracted using GEMMA. Further details on the triplet construction are provided in Appendix J.

For the comparison of the triplets, we develop three metrics: Event Coverage, Hallucination Rate, and Entity Accuracy.⁸

Event Coverage measures the proportion of gold events G_τ that can be matched to an extracted triplet from the generated text τ . For an event to be considered a match $m \in M_\tau$, all three components of the triplets must correspond to the gold event⁹

$$\text{EventCoverage}(\tau) = \frac{|M_\tau|}{|G_\tau|} \quad (3)$$

Hallucination Rate, on the other hand, measures the proportion of the generated text’s extracted triplets P_τ for which no corresponding gold event can be found.

$$\text{HallucinationRate}(\tau) = \frac{|P_\tau| - |M_\tau|}{|P_\tau|} \quad (4)$$

We additionally use Entity Accuracy to evaluate the three triplet components individually. For each component c , we define $M_{\tau, \setminus c}$ as the set of matches based on the other two components alone, and then check how many of these also match on c .

$$\text{EntityAccuracy}_c(\tau) = \frac{|M_\tau|}{|M_{\tau, \setminus c}|} \quad (5)$$

Further, we compare the language of the generated and the kicker texts using stylometric measures, which are explained in Appendix J.2.

5.2 Results

We evaluate texts generated from the Content Selection gold events (CS Gold) to isolate Text Generation performance, as well as reports generated with our full pipeline (FP) to assess its overall performance. All reports generated by our pipeline are available in our GitLab-Repository, with an example in Appendix I.2. Due to limitations of our

evaluation method (as discussed in Appendix A), we evaluate the kicker gold articles to establish a reference point for each metric.

Appendix J.2 shows that Event Coverage decreases from 34.9% for the kicker gold texts to 21.4% for the CS Gold-texts and 16.6% for the FP-texts. The FP-texts have the highest Hallucination Rate, while the texts generated from Content Selection gold events achieve an even slightly lower rate than the gold kicker texts. The latter may result from a higher diversity of information (e.g. atmosphere, background info/non-NLGE parts) in the kicker gold texts, creating more noise for the LLM triplet extraction. The individual Entity Accuracies of CS Gold-texts are higher than those of the gold kicker articles, likely because these entities are expressed more explicitly in our generated texts.

Overall, the low Event Coverage might result mainly from the LLM failing to extract events as triplets rather than extracting them incorrectly.

On top of that, we find stylistic differences that make the generated texts easily distinguishable from the gold texts. The results are further discussed in Appendix J.2.

6 Conclusion

We investigate whether it is possible to generate kicker-like football articles from structured match data. We develop a two-step Data-to-Text pipeline for automatic match report generation from football match data, split into Content Selection and Text Generation. We find that while our BiLSTM substantially learns which information is relevant for game reports, this is not always a clear-cut decision. Building upon the Content Selection, we generate natural language match reports from these selected events. GEMMA is capable of generating articles that superficially resemble kicker articles, but struggles with event coverage and stylistic variability. A key limitation is that the IMPECT-events do not represent all the information contained in a real kicker report. Consequently, even a strong Content Selection model cannot provide the Text Generation model with all the information typically available to a human sports journalist.

⁸Each metric is only defined for denominators > 0

⁹With a tolerance of ± 1 minute for the minute component.

References

- Leo Breiman. 2001. Random forests. *Machine learning*, 45:5–32.
- Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. 2021. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650.
- Stefan Evert, Thomas Proisl, Thorsten Vitt, Christof Schöch, Fotis Jannidis, and Steffen Pielström. 2015. Towards a better understanding of burrows’s delta in literary authorship attribution. In *Proceedings of the Fourth Workshop on Computational Linguistics for Literature*, pages 79–88, Denver, Colorado, USA. Association for Computational Linguistics.
- Gemma-Team. 2026. Gemma 4 technical report. *Preprint*, arXiv:2607.02770.
- Yongwan Gim and Kyushik Min. 2023. Evaluation strategy of time-series anomaly detection with decay function. *Preprint*, arXiv:2305.09691.
- Shahriar Golchin and Mihai Surdeanu. 2024. Time travel in LLMs: Tracing data contamination in large language models. In *The Twelfth International Conference on Learning Representations*.
- Alex Graves and Jürgen Schmidhuber. 2005. Frameworkwise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks*, 18(5-6):602–610.
- Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.
- Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization. *Preprint*, arXiv:1412.6980.
- Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.
- Gavin Leech, Juan J. Vazquez, Niclas Kupper, Misha Yagudin, and Laurence Aitchison. 2024. Questionable practices in machine learning. *Preprint*, arXiv:2407.12220.
- Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919.
- Franco Moretti. 2014. ‘conjectures on world literature’. *New Left Review*, 1:54–68.
- Marcel Nawrath, Agnieszka Nowak, Tristan Ratz, Danilo C. Walenta, Juri Opitz, Leonardo F. R. Ribeiro, João Sedoc, Daniel Deutsch, Simon Mille, Yixin Liu, Lining Zhang, Sebastian Gehrmann, Saad Mahamood, Miruna Clinciu, Khyathi Chandu, and Yufang Hou. 2024. On the role of summary content units in text summarization evaluation. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 272–281. Association for Computational Linguistics.
- J. O’Sullivan. 2025. Stylometric comparisons of human versus AI-generated creative writing. *Humanities and Social Sciences Communications*, 12:1708.
- Ehud Reiter and Robert Dale. 2000. *Building natural language generation systems*. Cambridge university press, Cambridge, UK.
- Rebecca Salles, Janio Lima, Michel Reis, Rafaelli Coutinho, Esther Pacitti, Florent Massegli, Reza Akbarinia, Chao Chen, Jonathan Garibaldi, Fabio Porto, and Eduardo Ogasawara. 2024. Softed: Metrics for soft evaluation of time series event detection. *Computers amp; Industrial Engineering*, 198:110728.
- Mandar Sharma, Ajay Kumar Gogineni, and Naren Ramakrishnan. 2024. Neural methods for data-to-text generation. *ACM Trans. Intell. Syst. Technol.*, 15(5).
- Anthony Sicilia, Konstantinos Pelechrinis, and Kirk Goldsberry. 2019. Deephoops: Evaluating micro-actions in basketball using deep feature representations of spatio-temporal data. *Preprint*, arXiv:1902.08081.
- Chris Van der Lee, Emiel Krahmer, and Sander Wubben. 2017. PASS: A Dutch data-to-text system for soccer, targeted towards specific audiences. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 95–104, Santiago de Compostela, Spain. Association for Computational Linguistics.
- Sam Wiseman, Stuart M. Shieber, and Alexander M. Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263, Copenhagen, Denmark. Association for Computational Linguistics.
- Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 207–212, Berlin, Germany. Association for Computational Linguistics.

A Limitations

Several limitations should be considered when interpreting the results of this project. These concern the available data, the model development and selection process, as well as the Text Generation and evaluation stages.

A.1 Dataset Construction

The entire annotation process (Text-to-NLGE, NLGE-to-Event(s)) was done by hand, with the data split between the authors. We agreed on rules and guidelines in advance and in unclear cases confided in one another to reach a unanimous decision. However, since the data was divided among annotators rather than being independently annotated by all of them, there remains a considerable risk of inconsistent annotations.

A.2 Data Limitations

One limitation of this project is the quality and completeness of the underlying match data. During dataset construction and annotation, we encountered inconsistencies and missing information in the structured event data. To document these issues, we introduced a flagging system during annotation. These flags marked different types of data problems, such as missing event information and inconsistencies in player or event timing. The flagging system mainly served to document issues, so that some data quality problems may still have affected both the gold annotations and the Content Selection experiments. The makeup of the faults in our data can be seen in figure 1:

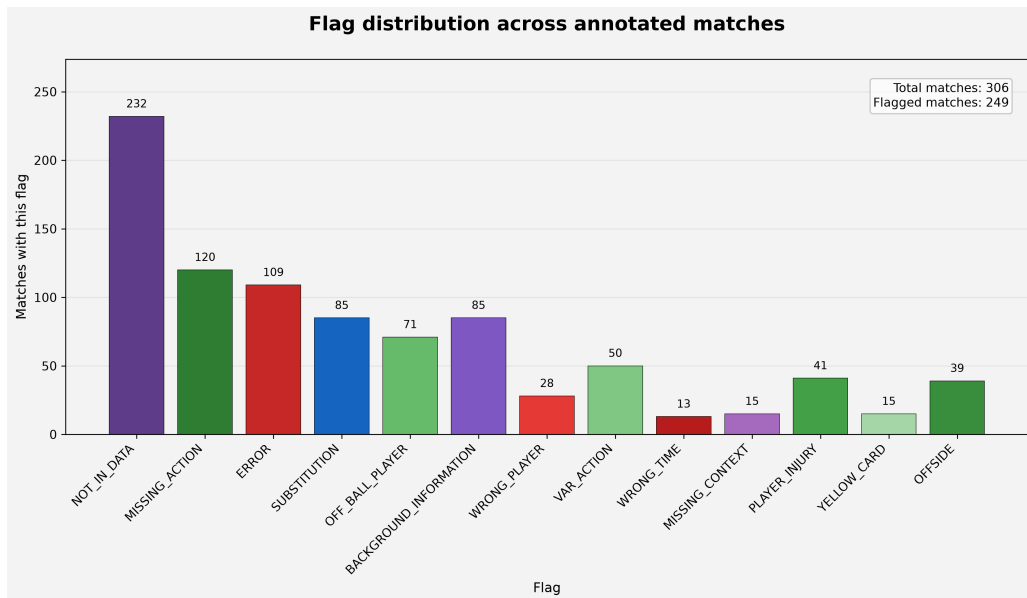


Figure 1: flag distribution

81% of match data contains at least one type of flag. Most faults accumulate towards context that is either inherently unquantifiable or just simply not in our data (see purple bars) because it references previous seasons, other competitions or things happening during the game but not on the field or independent of the ball (see green bars). Therefore we do not strive to match a kicker article in its entirety but rather only the parts that describe the actual game. Even so, we have to accept partly erroneous data as the basis of our project. To slightly mitigate this issue, we are able to aggregate all substitutions from non-event IMPECT data to solve one fault (see blue bar).

In addition, the version of the IMPECT-data available for the 2023-24 season contained less information than more recent versions of the dataset. Some features (e.g. woodwork) that could potentially be useful for Content Selection were added in later seasons, and therefore not available for our experiments. As a

result, some potentially useful information was unavailable not only for identifying report-worthy events, but also describing these events during Text Generation.

A.3 Content Selection

Our hyperparameter tuning was constrained in several ways. Due to limited computational resources, we tuned hyperparameters on a single cross-validation split (split 0) rather than training and evaluating each parameter combination across all nine splits. This limits the reliability of the selected combination and may not generalize across folds.

Additionally, the BiLSTM grid search was conducted in two rounds, meaning that the hyperparameter combinations explored in the second round were centered around the values fixed in the first round. A better approach might have been a random search.

The permutation feature importance results suggest that our initial manual feature selection was not ideal, as we systematically underestimated the relevance of positional features for our classification task (see Section 4.2.2). This assumption is further supported by comparison of the BiLSTM trained on the full feature selection and the BiLSTM trained on our manual selection (Section 4.2). This, however, was also only explored on split 0 (again, for computational reasons).

A.4 Text Generation

The text-generation is zero-shot-prompting, which may not be ideal, as multiple rounds of prompting may increase coverage of events in the generated texts or reduce hallucinations. Also, because of our tech-stack based on llama-cpp-python, we are limited to models with the GGUF-File-format, which could be avoided through a rewrite of the project.

The triplet-based evaluation represents events using minute, player ID and action type components. However, these components are not always explicitly stated in match reports. If an event is described without an exact minute or player, or the action is implied rather than mentioned, the event cannot be extracted as a complete triplet and therefore cannot be matched to a gold event.

Another limitation is that triplets from generated texts are extracted using an LLM. This means that the evaluation depends not only on the quality of the generated report, but also on how accurately events can be converted into triplets. If an event is not described explicitly, it may not be extracted correctly. As a result, a missing or unmatched triplet in Event Coverage or Hallucination Rate does not always indicate an error in the generated report itself, but can also occur due to limitations in the extraction process. Since there is no certain metric to measure factual consistency between gold and generated texts, the triplet-based approach provides a systematic way to make this comparison. However, the possible influence of the extraction process should also be considered when interpreting the evaluation results.

A.4.1 Contamination

Our contamination evaluation is currently limited to testing for factual parametric knowledge in the LLMs. However, recent work has also explored checking for verbatim memorization of specific corpora (for us, this would be the kicker articles), which we do not test for in our models. Additionally, when automatically evaluating GEMMA’s answers, using only exact string matching for player names might be too strict and could lead to an underestimation of its inherent knowledge. Furthermore, parts of the evaluation were useless, as the model did not predict any red cards and did not produce any valid output for six of the 306 tested matches.

B Manual Selection Rules

Rule	Explanation
(Own)Goal Rule	Always include goals
Red Card Rule	Always include red card events
Penalty Rule	Always include penalty events (and the following event)
PxT-Shot Rule	Include shots which have a PxT-Metric ¹⁰ higher than 0.05 and the following event
Shot-in-Box Rule	A shot originating inside box will be selected with the following event
Corner and Shot	If a corner results in a shot inside the box

Table 4: Overview of the manually defined selection rules

C Random Forest Training Details

Hyperparameter	Values searched	Best (selected)	Best (full)
Number of trees (n_estimators)	100, 300, 500	300	500
Class weight	None, balanced, {0:1, 1:10}, {0:1, 1:50}, {0:1, 1:100}	{0:1, 1:100}	{0:1, 1:10}
Max. leaf nodes	None, 50, 100, 250	None	None
Max. samples per tree	None, 0.3	0.3	0.3

Table 5: Random Forest grid search space ($3 \times 5 \times 4 \times 2 = 120$ configurations per feature set) and the best configuration selected via dev set binary F1, for split 0

D BiLSTM Training Details

Hyperparameter	Setting
Loss	BCEWithLogitsLoss
Optimizer	Adam
LR scheduler	ReduceLROnPlateau (factor=0.5, patience=2, min_lr=1e-6)
Gradient clipping	max_norm=1.0
Early stopping	patience=3 (on dev F1)
Prediction threshold	logit > 0
Seed	33
Batch size	64
Epochs (max)	20
Dropout	0.1
Batch size	64
Epochs (max)	20

Table 6: Training configuration for the BiLSTM classifier

E BiLSTM Grid Search Details

Hyperparameter	Values searched	Best (selected)	Best (full)
Embedding dim	8, 16, 32	32	32
Hidden dim	128, 256, 512	256	128
Num. layers	1, 2	2	2
Learning rate	3e-4, 1e-3	1e-3	1e-3

Table 7: BiLSTM grid search space, round 1 ($3 \times 3 \times 2 \times 2 = 36$ configurations per feature set; window size fixed at 31, neg:pos ratio fixed at 75) and the best configuration selected via dev set binary F1, for split 0

¹⁰A metric of the IMPECT-Dataset which measures the goal threat a team currently possesses

Hyperparameter	Values searched	Best (selected)	Best (full)
Window size	1, 11, 21, 31, 41	31	31
Neg:pos ratio	45, 55, 65, 75, 85, 95	75	65

Table 8: BiLSTM grid search space, round 2 ($5 \times 6 = 30$ configurations per feature set; embedding dim, hidden dim fixed, num. layers, learning rate fixed at the respective values found in grid search round 1) and the best configuration selected via dev set binary F1, for split 0

F Feature Selection

Table 9 contains an overview of the full (44 features) and manual (22 features) feature selection with justifications for the manual feature selection decisions. These justifications are based on assumptions we initially made about the relevance of features and they do not fully align with our later findings on permutation feature importance.

Table 9: Overview of all event features considered and the resulting feature selection

Feature	Full Feat.	Manual Selection	Justification
Numerical			
pressure	✓	✓	Implies intensity of event, more intense events might be more relevant.
duration	✓	✓	Sequences of many events with low duration may imply relevant, high intensity phases during the match. Events with exceptionally long duration might be outstanding dribbles, solo runs, etc. by one player. On the other hand, goals and fouls are also often assigned a high duration because of the consequent break in play.
opponents	✓	✓	Describes the number of players able to defend their own goal. Number of opponents could therefore imply threat of current position.
distanceToGoal	✓	✓	Events closer to either goal might be more relevant.
gameTimeInSec	✓	✓	Time of event during the match might add to its relevancy, e.g. the last attack of each game towards the end will maybe be mentioned in an article with higher likelihood.
gameTime	✗	✗	Game time in string-Format. Already covered by gameTimeInSec.
start_x, start_y, end_x, end_y	✓	✓	Location of event might correlate with relevancy. Covered distance (difference between start to end) might imply progression of an attack.
start_adj_x, start_adj_y, end_adj_x, end_adj_y	✓	✗	Not selected in favor of the raw coordinates, might have been more suitable for an ML-model.

Continued on the next page

Table 9 – Continuation

Feature	Full Feat.	Manual Selection	Justification
pxT_team, pxT_opponent	✓	✓	Events with high expected goal threat will probably be considered more relevant.
pass_distance, pass_angle	✓	✗	Exists only for actions with attribute pass and would therefore mostly be left blank in a context window with mixed events. Could also produce a misleading signal due to our preprocessing, as missing numerical values are set to 0.0.
shot_distance, shot_angle, shot_targetPoint_y, shot_targetPoint_z	✓	✗	Exists only for actions with attribute shot and would therefore mostly be left blank in a context window with mixed events. Could also produce a misleading signal due to our preprocessing, as missing numerical values are set to 0.0.
Categorical			
action, actionType	✓	✓	Crucial for determining relevancy of an event. For some actions, we ideally expect classification as relevant in all cases (e.g. goals, red cards).
result	✓	✓	Success or failure of an event probably influences relevancy.
phase	✓	✓	May not only influence classification of current, but also classification of surrounding events, for example if an event is part of an attack through set piece.
bodyPart, bodyPartExtended	✓	✓	Body part may correlate with relevancy, for example headers might be considered more spectacular and therefore more relevant.
previousPassHeight	✓	✓	May provide direct insight into the pace of the game and the risk involved in an offensive play. High passes can bypass the opposing defense while low passes stand for combination play.
distanceToOpponent	✓	✓	Gives insight about under how much pressure the player currently holding the ball is.
periodId	✓	✗	Game period (first half, extra time, etc.) may be insightful, but is also already implied by gameTimeInSec.
inferredSetPiece	✓	✗	Technical information, probably not relevant for our classification task.
player_position, player_positionSide	✓	✓	While the player ID is omitted, we use the combination of player_position and player_positionSide to provide the model with tactical context and an understanding of player's roles.

Continued on the next page

Table 9 – Continuation

Feature	Full Feat.	Manual Selection	Justification
start_packingZone, end_packingZone	✓	✗	Might be unnecessarily fine-grained for the classification task. Might also be somewhat redundant to pitch position and coordinates.
start_pitchPosition	✓	✓	Location of event might correlate with relevancy. This information might already be covered by the start coordinates, but might also provide this information in a way that is more suitable for the model.
end_pitchPosition	✓	✗	Should have been included for design-consistency (because start_pitchPosition was also included), but on the other hand might also already be covered by the end coordinates.
start_lane, end_lane	✓	✗	Horizontal zone of an event probably does not heavily correlate with its relevancy for a football article.
pass_receiver	✓	✗	Exists only for actions with attribute pass and would therefore mostly be left blank in a context window with mixed events. Also, the information is included in the following event.
duel_duelType	✓	✗	Exists only for actions with attribute duel and would therefore mostly be left blank in a context window with mixed events. One could argue that it could otherwise be useful information.
formation_team, formation_opponent	✓	✗	Formation stays mostly the same throughout a match and will therefore probably be mostly redundant. Might also be more interesting to state in the introduction of a football article, but not directly relevant to our classification task.
setPiece_mainEvent	✓	✗	Set pieces are usually characterized by specific action types anyway (like free kick, corner, etc.), so this binary flag probably only provides redundant information.

ID-Features

id, squadId, player _i d, duel_player pass_receiver_player currentAttackingSqu pressingPlayerId, fou dribble_playerId, set setPiece_subPhaseId	✗	✗	Does not naturally correlate with our classification task, and may therefore be a harmful signal or lead to skewed results. Also risk of overfitting.
index, SequenceIndex	✗	✗	Relative counter that is not comparable among matches.

Continued on the next page

Table 9 – Continuation

Feature	Full Feat.	Manual Selection	Justification
Additional features (dataV4+ / seasons starting from summer 2024)			
shot.gk (coordinates, adjCoordinates, divePoint)	X	X	According to the documentation, this is only available starting with dataV4. In our dataset, this feature is not available.
shot.woodwork	X	X	According to the documentation, this is only available starting with dataV4. In our dataset, this feature is not available.
opponent (Top-Level: coordinates, adjCoordinates)	X	X	According to the documentation, this is only available starting with dataV4. In our dataset, this feature is not available.
dribble (distance, type, result)	X	X	According to the documentation, this is only available starting with dataV4. In our dataset, this feature is not available.

E.1 Preprocessing of numeric and categorical features

For the Random Forest, numeric features are not normalized and missing values are replaced with 0. Categorical features are one-hot encoded. We then combine the numeric features and the one-hot-encoded categorical features into a single feature vector.

For processing the events in the BiLSTM, we z-normalize the numeric features (the mean and standard deviation are determined on the training dataset), while the categorical features are transformed via label encoding (unknown values receive the <UNK>-token).

As we do not want to assume any ordinality of the categorical features, these encoded labels are not projected directly into a continuous vector space, but rather via separate embedding layers in the model. Thus there exists an embedding-layer for each feature in the BiLSTM that can be trained together with the rest of the model. The feature vectors of the previous and following events are then combined into a context window, with the event to be classified in the middle.

F.2 Feature Importance

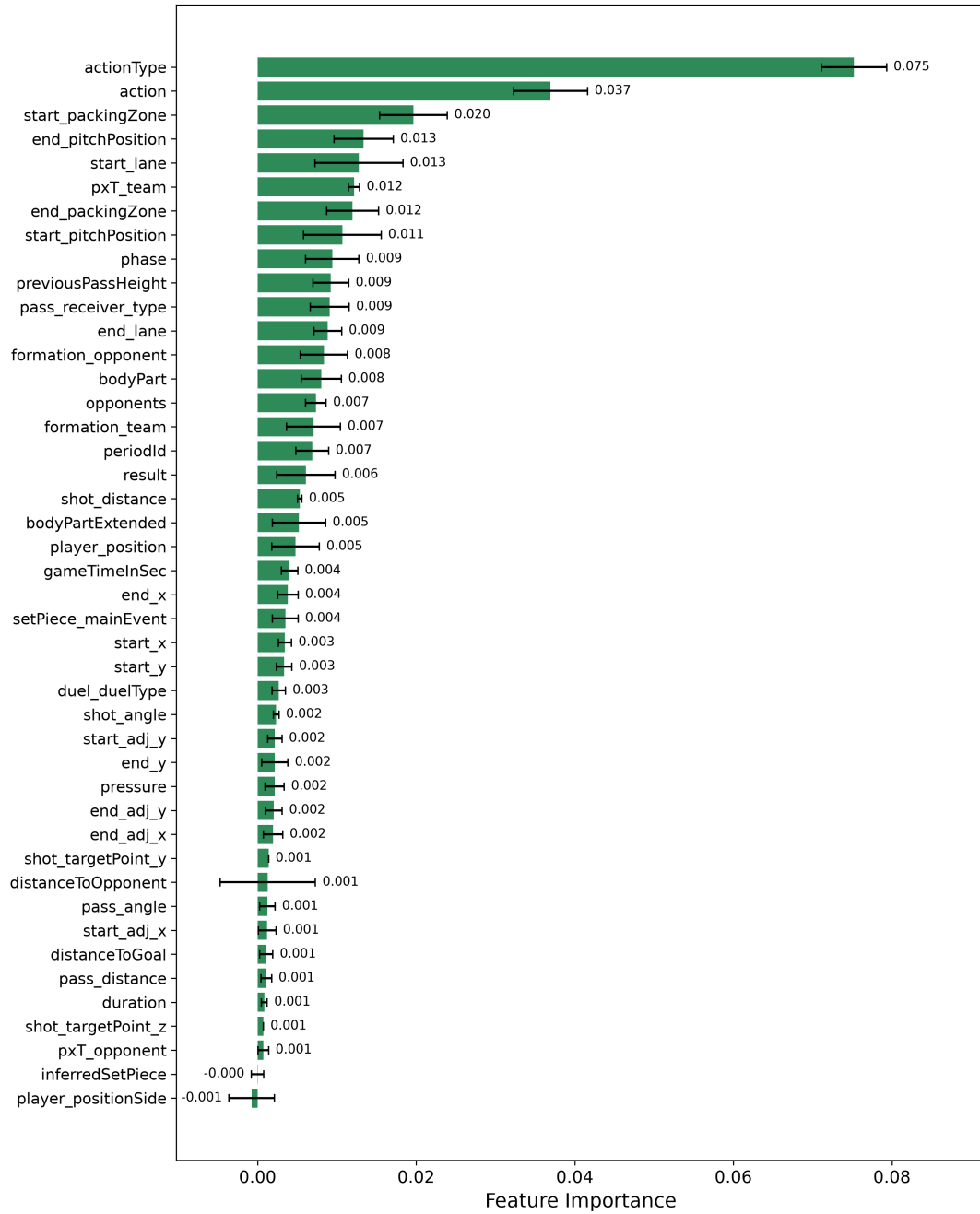


Figure 2: Permutation Feature Importances (BiLSTM trained on Full Selection.)

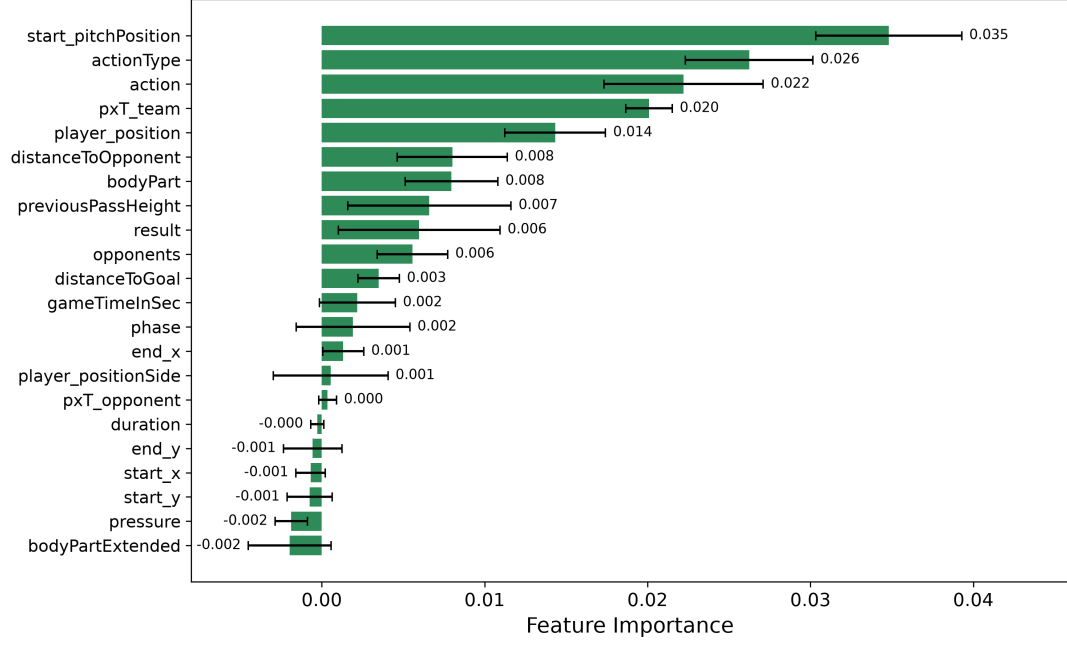


Figure 3: Permutation Feature Importances (BiLSTM trained on Manual Selection).

G Content Selection Results

Model	0	1	2	3	4	5	6	7	8	p (vs. previous)
Rule-Based	0.460	0.460	0.460	0.460	0.460	0.460	0.460	0.460	0.460	–
RF (selected)	0.528	0.528	0.529	0.526	0.523	0.518	0.526	0.517	0.526	<0.001
RF (full)	0.542	0.540	0.538	0.540	0.542	0.537	0.539	0.544	0.544	0.03
BiLSTM (selected)	0.607	0.591	0.570	0.585	0.589	0.603	0.599	0.607	0.592	<0.001
BiLSTM (full)	0.614	0.604	0.618	0.601	0.612	0.614	0.586	0.620	0.613	0.217

Table 10: Test set F1 per split. The rightmost column reports the paired bootstrap significance (p) of each model outperforming the model in the row directly above, computed on the split 0 test set.

Model	0	1	2	3	4	5	6	7	8
Rule-Based	0.480	0.480	0.480	0.480	0.480	0.480	0.480	0.480	0.480
RF (selected)	0.560	0.558	0.560	0.554	0.553	0.549	0.560	0.548	0.556
RF (full)	0.573	0.571	0.565	0.570	0.569	0.565	0.571	0.574	0.574
BiLSTM (selected)	0.652	0.625	0.593	0.609	0.626	0.640	0.652	0.645	0.624
BiLSTM (full)	0.647	0.634	0.664	0.643	0.646	0.650	0.614	0.665	0.650

Table 11: Test set weighted F1 (wF1) per split.

G.1 BiLSTM Hyperparameters

F1 Score per Downsampling Ratio, averaged across Window Sizes [1, 11, 21, 31, 41]

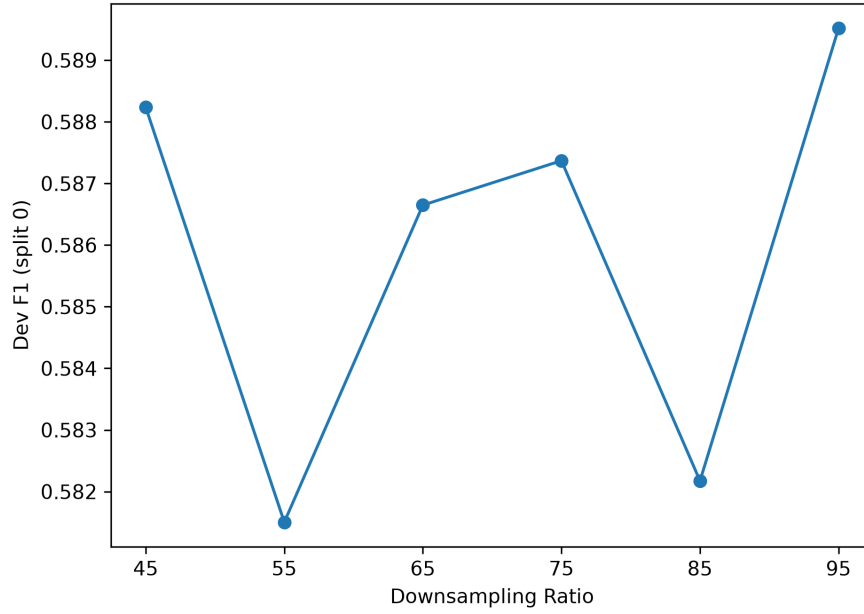


Figure 4: Dev F1 (Split 0) of selected-feature BiLSTM as a function of downsampling ratio, averaged across all tested window sizes

F1 Score per Window Size, averaged across Downsampling Ratios [45, 55, 65, 75, 85, 95]

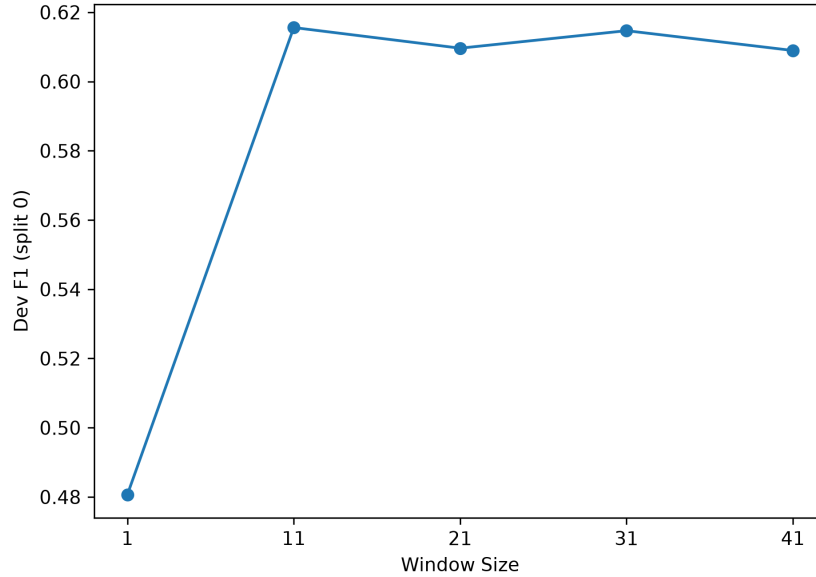


Figure 5: Dev F1 (Split 0) of selected-feature BiLSTM as a function of window size, averaged across all tested downsampling ratios

G.2 BiLSTM Error Analysis

actionType / result	TP	FP	FN	TN	Recall	Prec.
PASS / FAIL	9	4	25	5349	0.26	0.69
PASS / SUCCESS	42	25	46	20511	0.48	0.63
SHOT / FAIL	113	68	120	417	0.48	0.62
SHOT / SUCCESS	101	2	0	0	1.00	0.98

Table 12: Recall and precision by result

actionType / distanceBucket	TP	FP	FN	TN	Recall
SHOT / 0–10	70	32	20	49	0.78
SHOT / 10–20	125	29	69	195	0.64
SHOT / 20–30	17	8	29	152	0.37
SHOT / 30–40	1	1	2	18	0.33
RECEPTION / 0–10	27	20	10	142	0.73
RECEPTION / 10–20	22	16	12	603	0.65
RECEPTION / 20–30	8	8	8	1376	0.50
RECEPTION / 30–40	2	7	4	2791	0.33

Table 13: Recall by distance to goal (in meters) for SHOT and RECEPTION

actionType	TP	FP	FN	TN	Recall
OUT	1	10	64	1890	0.02
BLOCK	1	4	28	1175	0.03
GK_CATCH	0	0	19	425	0.00
OFFSIDE	0	1	9	72	0.00

Table 14: Action types with consistently low recall (≤ 0.03)

actionType / action	TP	FP	FN	TN	Recall	Prec.
GOAL	102	1	0	0	1.00	0.99
OWN_GOAL	1	0	0	0	1.00	1.00
SHOT / PENALTY_KICK	12	0	0	0	1.00	1.00
FREE_KICK / SUCCESS	2	0	0	492	1.00	1.00
SHOT / ONE_VS_ONE_AGAINST_GK	21	2	0	1	1.00	0.91

Table 15: Action types with consistently perfect recall (1.00)

H Contamination

H.1 Methods

Data contamination and memorization can diminish the reliability of the evaluation of LLMs (Carlini et al., 2021; Golchin and Surdeanu, 2024). Additionally, for our experiments even parametric knowledge of the LLM used for text generation about the Bundesliga 2023-24 matches could prove to be harmful to an unbiased evaluation of our method and generated texts. The model relying on its own knowledge to enhance the texts or add missing information would then lead to unfaithful generations and extrinsic hallucinations (Maynez et al., 2020; Ji et al., 2023) and thus inflated performance. We therefore control for parametric knowledge of 300¹¹ games in the 2023–24 Bundesliga season for GEMMA by prompting regarding two properties (see Appendix H.2 for the full prompts):

¹¹For reasons unbeknownst to us, GEMMA did not produce valid output for the remaining 6 matches.

- (1) For each goal in the game, list the scorer and corresponding minute of the match.
- (2) For each red card issued during the game, list the penalized player and the corresponding minute of the match.

We then automatically compare GEMMA answers to our gold data (players-only, minutes-only, player-minute pairs per property). If the model is able to reproduce many facts correctly, we consider it contaminated and not suitable for our task.

Additionally, we manually check a sample for GPT-5.6 (Appendix H.3).

H.2 LLM Prompts for Parametric Knowledge Evaluation

The following prompts are used to investigate GEMMA's (and GPT-5.6 SOL's) inherent parametric knowledge of Bundesliga season 2023-24 matches.

- **Goals Prompt:**

"Nenne mir alle Torschützen des Spiels team1 gegen team2 vom x. Spieltag der Bundesliga-Saison 2023-24 mit Vor- und Nachnamen und jeweils die Spielminute, in der sie getroffen haben im Format MM (z.B. 08). Sollte ein Tor in der Nachspielzeit einer der beiden Halbzeiten gefallen sein, dann bezeichne diesen Fall bitte immer einfach mit '45+' oder '90+'. Es ist auch möglich, dass keine Tore gefallen sind."

- **Red Cards Prompt:**

"Nenne mir alle Spieler des Spiels team1 gegen team2 vom x. Spieltag der Bundesliga-Saison 2023-24 mit Vor- und Nachnamen, die eine rote Karte bekommen haben und jeweils die Spielminute, in der sie die rote Karte bekommen haben im Format MM (z.B. 08). Sollte dies in der Nachspielzeit einer der beiden Halbzeiten geschehen sein, dann bezeichne diesen Fall bitte immer einfach mit '45+' oder '90+'. Es ist auch möglich, dass keine roten Karten vergeben wurden."

H.3 Sample Inspection of GPT

To test whether GPT-5.6 SOL had prior knowledge of the matches, we conduct a contamination test on the Werder Bremen vs. FC Bayern München match from the first match day and manually evaluate the model's response. The test is performed without intermediate reasoning steps and without providing the model with access to the Internet or any match data. No temperature value is set and the model's default setting is used. The model is able to correctly provide information about the asked events, which indicates potential contamination.

H.4 Evaluation of GEMMA

To assess GEMMA's knowledge of the Bundesliga season 2023-24 games, its responses were automatically checked against data based on the aggregated info from the IMPECT data¹² on three independent levels with precision, recall and F1 score:

1. **Player-only:** Evaluate only names of players involved, consider them as independent from minute counts. Only exact string matches count as correct.

Example: Gemma: Messi - 10' vs. Gold: Messi - 78'
→ **Outcome:** 100% Precision / Recall / F1 (Player name matches exactly).

2. **Minute-only:** Evaluate only minute counts, consider them as independent from the player names. Minute counts do not have to be an exact match, as we noticed some deviation even in our gold data when comparing to external sources (such as the kicker-Liveticker). Minute counts therefore count as a match if they do not have a difference greater than three minutes. Extra time is just marked with a "+" ("45+", "90+") and counts as a match if there is an exact (sub)string match (e.g. "90" and "90+").

Example: Gemma: Messi - 10' vs. Gold: Ronaldo - 11'
→ **Outcome:** 100% Precision / Recall / F1 (Minute difference ≤ 3 min).

¹²<https://github.com/ImpectAPI/open-data>

3. **Player-minute:** Evaluate player names and minute counts as pairs, only count an instance as a true positive if both player name and minute count match (with the above stated matching methods).

Example: Gemma: Messi - 10' vs. Gold: Messi - 11'
→ **Outcome:** 100% Precision / Recall / F1 (Both player and minute match).

H.4.1 Experimental Setup & Automatic Evaluation

For the parametric knowledge experiments, we evaluate the 4-bit quantized version (Q4_0) of Gemma-4-31B-it. To ensure deterministic and therefore reproducible outputs, we set the generation temperature to 0.0 (greedy decoding) with a maximum token limit of 10,000. Additionally, internal reasoning mechanisms are disabled. To make automatic evaluation easily possible, we constrain GEMMA’s responses to a fixed JSON-schema (structured output) containing the player names and minutes.

We compute micro- and macro average for precision, recall and F1 score. To avoid unnecessarily penalizing the model and thus underestimating its knowledge when macro-averaging, we assign a score of 100% for precision, recall and F1 score, if there were no events in the given event category (e.g. no goals throughout the match, final score of 0:0), and correctly gave an empty response.

H.4.2 GEMMA Contamination Evaluation Results

Table 16 shows our findings when automatically checking GEMMA’s responses against the ground truth from the IMPECT data for 300 Bundesliga season 2023-24 matches, and Table 17 shows additional evaluation statistics.

Category	Level	Micro Average			Macro Average		
		Prec.	Rec.	F1	Prec.	Rec.	F1
Goals	Player-only	0.104	0.127	0.114	0.102	0.128	0.106
	Minute-only	0.206	0.253	0.227	0.197	0.253	0.204
	Player-minute	0.019	0.023	0.021	0.023	0.030	0.024
Red Cards	Player-only	0.000	0.000	0.000	0.850	0.850	0.850
	Minute-only	0.000	0.000	0.000	0.850	0.850	0.850
	Player-minute	0.000	0.000	0.000	0.850	0.850	0.850

Table 16: Contamination Evaluation Results ($N = 300$ matches).

Property	Observed Value
<i>Dataset Overview</i>	
Total Evaluated Matches	300
Perfectly Known Games	2
<i>Goals</i>	
Average Goal Deviation (Pred vs. Gold)	1.95
Average Goal Bias	0.7
Empty Responses (GEMMA / Gold)	17 / 13
<i>Red Cards</i>	
Empty Responses (GEMMA / Gold)	300 / 255

Table 17: Overall Match Statistics for Parametric Knowledge Evaluation ($N = 300$).

When looking at the goals event category in Table 16, with a Micro F1 of 11.4%, GEMMA does perform considerably better than a random baseline (random guessing of player names). The model generally seems to know which players belong to which team (after all, we only gave the team name and not a list of candidate players in the prompt) and possibly also which players are more likely to score a goal (e.g., strikers compared to the goalkeeper). One could argue that this knowledge is probably not harmful in our pipeline, as it resembles sports reporters writing football articles and can therefore even be considered desirable. Nevertheless, this performance does not suggest that GEMMA reliably memorizes goal scorers for specific games.

Examining the minute-only level, we observe even higher performance with a Micro F1 score of 22.7%. We partially attribute this to the tolerance window of ± 3 minutes, which makes it more likely for the model to guess correctly by chance. Nonetheless, this suggests real (albeit limited) parametric knowledge about the temporal progression of the games. Generally, recall is a bit higher than precision, as GEMMA predicts 0.7 more goals per game on average than ground truth (Table 17).

However, looking at the player-minute scores, we find that GEMMA does not actually perform well (2.1% Micro F1). Even if GEMMA knows about two percent of goals perfectly from its parametric knowledge, we do not consider this a high enough level of contamination to be harmful to the validity of our text generation results.

Interestingly, GEMMA reaches "perfect knowledge" for 2 out of 300 games (Table 17). Matches are considered as "perfectly known", if there are no false positives and no false negatives for both the goals and the red card event category. Upon further investigation we find that both of these matches do not contain any goals or red cards. Therefore, we do not consider these two perfectly known games as an indicator for contamination.

Unfortunately, when provided with our prompt (see Appendix H.2), GEMMA does not predict any red cards for any of the 300 games. This might be due to the phrasing of the prompt, some technical issue, etc. and does not necessarily show that GEMMA is not contaminated. Unfortunately, we have to consider this an invalid signal and cannot use it for our contamination evaluation. When Macro-averaging with the method stated above, this leads to "surprisingly" high (but not meaningful) scores (Table 16), as the majority of football matches actually come without any red cards.

I Text Generation Prompts

I.1 Text Generation Prompt¹³

```
Du bist ein Sportreporter mit Expertise in der ersten Bundesliga und du wurdest damit beauftragt, unterhaltsame schriftliche Spielzusammenfassungen zu schreiben. Du bekommst eine JSON-Datei mit einer Liste von Spielzügen eines Bundesligaspiels, die aus einem Datenset stammen und weiteren Informationen. Die Antwort muss auf Deutsch sein. Die Team und die Spielernamen können über die ID per Lookup in der "player_mapping" oder "team_mapping"-Eigenschaft der JSON aufgelöst werden. Die Daten haben zur Hilfestellung noch Auswechselungen, alle Tore, der Endstand und roten Karten nochmal explizit hinzugefügt. Nutze diese Explizit, um zu überprüfen, ob der Endstand richtig ist, und ob zu den einzelnen Toren die Daten (Minute, Torschütze) stimmen. Füge diese nicht verbatim am Ende hinzu, sondern baue sie natürlich in den Text ein. Auswechselungen sollen nicht alle erwähnt werden, nur die, die außergewöhnlich früh sind (z.B. in der ersten Halbzeit), oder welche, die das Momentum des Spiels verändert haben. Beispiel für JSON – es werden nur die wichtigsten Attribute erwähnt:
```json
{
 "events": [<event-daten>],
 "player_mapping": [{"id": <spielerid>, "firstname": <firstname>, "lastname": <lastname>}],
 "team_mapping": [{"id": <teamid>, "name": <teamname>, "home": true/false, "id": <teamid>, "name": <teamname>, "home": true/false},
 "substitutions": [{"minute": <timestamp>, "team_id": <teamid>, "player_id": <spielerid>, "from_position": <pos>, "to_position": <pos>}],
 "goals": [{"minute": <timestamp>, "team_id": <teamid>, "player_id": <spielerid>, "score": <score>}],
 "red_cards": [{"minute": <timestamp>, "team_id": <teamid>, "player_id": <spielerid>}],
 "score": <score>
}
```
```

¹³reproduced verbatim

I.2 Example Generated Text

****Heidenheimer Aufholjagd: Drama in der Voith-Arena!****

Was für ein Spiel! Wer in der Halbzeitpause die Fernbedienung beiseitegelegt hat, hat eines der spektakulärsten Comebacks der Saison verpasst. Der 1. FC Heidenheim dreht ein Spiel gegen den FC Bayern München, das eigentlich schon entschieden schien.

Die erste Halbzeit gehörte komplett den Münchner Gästen. Die Bayern agierten souverän und ließen dem Heidenheimer Schlusslicht kaum Luft. In der 38. Minute fiel dann die logische erste Antwort: Nach einer Kombination über die Flügel bediente Serge Gnabry den einen Harry Kane, der eiskalt aus etwa 12 Metern zum 0:1 vollstreckte. Doch die Bayern waren noch lange nicht fertig. Nur sechs Minuten vor dem Pausenpfeiff erhöhte Serge Gnabry selbst. Nach einer präzisen Flanke von Alphonso Davies stieg Gnabry im Zentrum hoch auf und drückte den Ball per Kopf zur 0:2-Führung in die Maschen. Mit einer 0:2-Niederlage in den Kabinen ging es für Heidenheim in die Pause – ein Ergebnis, das fast schon wie ein Todesurteil wirkte. Doch der zweite Durchgang wurde zu einer völlig anderen Geschichte.

Kurz nach dem Neustart, in der 50. Minute, entzündete Heidenheim den Funken der Hoffnung. Kevin Sessa tauchte nach einer schnellen Umschaltphase im Strafraum auf und schoss die Gastgeber auf 1:2. Dieses Tor veränderte die gesamte Dynamik des Spiels. Das Momentum war gekippt, und die Heidenheimer warfen alles nach vorne. Nur zwei Minuten später folgte der Schock für die Münchner: Tim Kleindienst profitierte von einem genialen chipped Pass von Niklas Beste und glich aus. 2:2! Die Voith-Arena bebte, die Bayern wirkten plötzlich orientierungslos.

Trotz einiger Versuche der Bayern, das Spiel wieder unter Kontrolle zu bringen – so sah Mathys Tel in der 68. Minute einen glänzenden Reflex von Heidenheims Kevin Müller – blieb der Druck der Gastgeber hoch.

Das Finale schrieb dann die dramatischste Wendung: In der 79. Minute gelang Heidenheim die sensationelle Führung. Marvin Pieringer flankte den Ball butterweich in den Strafraum, wo Tim Kleindienst erneut die richtige Position fand und den Ball im Eins-gegen-Eins am Torhüter vorbei zum 3:2-Endstand zirkelte.

Die Bayern versuchten zwar noch, das Spiel zu retten, doch Heidenheim hielt die Defensive dicht. Am Ende steht ein unglaubliches 3:2 für den 1. FC Heidenheim. Ein Sieg, der zeigt, dass in der Bundesliga niemals alles entschieden ist, bevor der Schlusspfeiff ertönt. Die Bayern reisen mit einem bitteren Geschmack nach München, während Heidenheim eine Party feiert, die man so schnell nicht vergessen wird.

Here is the original kicker article¹⁴.

J Text Generation Results

J.1 Triplet Construction

The following describes how the triplets used for the evaluation were constructed from the gold and generated texts.

For the gold triplets, the player ID and action type could be directly extracted from the data, since the gold events had already been selected based on the gold texts. However, the minute information required a different approach. The minute recorded in the data does not correspond exactly to the minute stated in the gold text. To avoid such inconsistencies during the evaluation, we used the minutes stated in the gold texts as the reference minutes for the gold triplets.

For the generated texts, the described events could not be directly linked to the structured event data. We therefore use GEMMA to extract the triplets. The model extracts the minute directly from the text, player ID using a mapping between player names and IDs, and the action type based on the action types defined in the data documentation.

Since LLM-based triplet extraction introduces an additional potential source of error, we first evaluate the reliability of the extraction process by applying it to the gold texts and comparing the extracted triplets with the gold triplets.

¹⁴<https://www.kicker.de/heidenheim-gegen-bayern-2024-bundesliga-4862217/analyse>

J.2 Factual Consistency Results

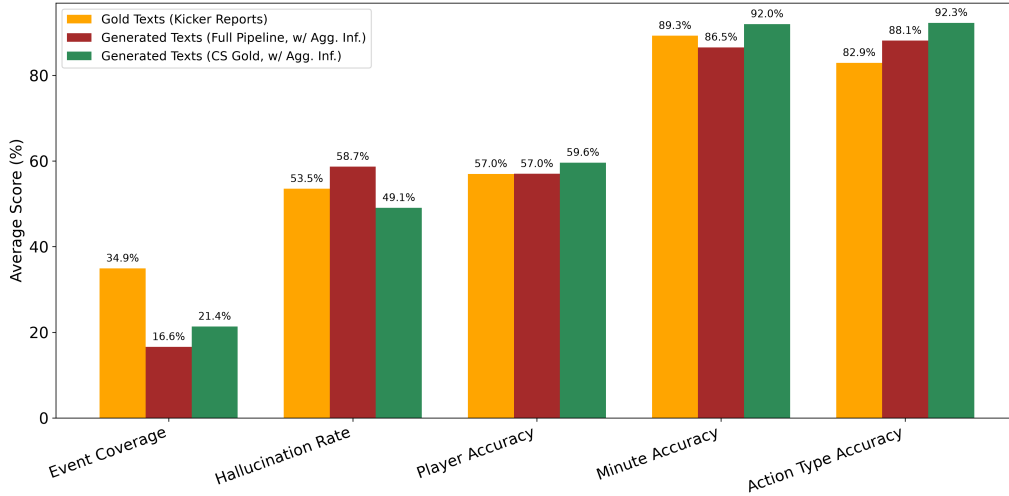


Figure 6: Comparison of the triplet-based evaluation results for gold texts and generated texts using BiLSTM and Gold Content Selection, with aggregated information, on the test set.

J.3 Style Evaluation

To stylistically evaluate large quantities of text we look to established Distant Reading (Moretti, 2014) approaches as guidance. We employ the Burrows’s Delta metric for authorship attribution, where we distinguish between kicker reports and our generated output. Further lexical examination includes vocabulary size and n-gram analysis while syntactical examination features sentence type and complexity. For analysis of the lexical domain we measure vocabulary size as it forms the base for all following inquiries. For n-gram analysis $n=4$ is typically chosen to avoid the most common, domain-unspecific linking phrases. In the following table our generated output is referred to as GEMMA. We evaluate an output of the 30 texts generated from the test set, matched to their corresponding kicker reports.

The syntactical domain is analyzed through sentence type and complexity. For both of these measures we first count the number of sentences by splitting them at the full stop which is flawed. However, for our use-case the actual number of sentences is negligible as we care about the relative comparison between the text types. The working assumption is that all kicker texts consist of 944 and all GEMMA texts consist of 728 sentences, making the difference in exclamation sentences ever so striking. The generated texts feature 37 exclamation sentences, compared to 1 exclamation in the kicker reports. This means that a GEMMA-output text averages at 1,2 exclamations per report. For sentence complexity we look for conjunctions that initiate relative clauses, as we assume relative clauses (nesting) to approximate complexity. We count 190 kicker and 98 generated sentences with relative clauses, the ratio being 20% vs 13%. It is important to note that some conjunctions like "da", "als" and "wie" act as words of comparison as well and can therefore be read as noisy for this specific task. However, comparisons make the text more complex as well, so the overall evaluation still rings true. If we assume LLM writing to be more sensational and less complicated than human writing (O’Sullivan, 2025), our findings suggest that our output is written akin to typical LLM fashion.

Table 18: Comparison of kicker and GEMMA texts

| text | kicker | GEMMA |
|----------------------|--------|-------|
| vocab size | 3260 | 1885 |
| unique n-grams (n=4) | 16480 | 9223 |
| Exclamations | 1 | 37 |

We infer that smaller vocabulary leads to less creative output, measurably seen in fewer unique n-grams. Those fewer n-grams are then repeated more often, making the output more predictable.

Burrow’s Delta is a measure for stylometric difference in literary authorship attribution (Evert et al., 2015). The Delta score for a document is the average of the absolute standardized differences across the most informative features:

$$\Delta_i = \frac{1}{n} \sum_{j=1}^n |z_i(j) - z_t(j)|$$

To calculate, we build a corpus-level feature matrix from the reference summaries. Our corpus consists of reports generated on the test set and their corresponding kicker articles. For each document, we compute its feature vector x . For each feature j , we compute the z-score-like deviation from the reference mean, using the absolute difference from the mean across the corpus. A score of 0 is interpreted as a perfect stylometric match, meaning the two texts are assumed to be written by the same author. Author in our case means either GEMMA-generated or kicker reports. Our average delta lies at 0.972 which implies noticeable stylometric differences. A GEMMA report is closer to another GEMMA report than to a kicker report, the inherent GEMMA delta being 0.838 and the inherent kicker delta being 0.823. This means that GEMMA has a consistent style of its own, distinct from kicker reports.

It is important to note that we did not finetune our model, which could have helped in stylistically aligning our output with the kicker reports. All our measures individually do not account for major differences themselves, in combination however, they imply a trend for noticeable stylistic differences. A human close reading and literal analysis (which is not part of the report) seem to confirm this hypotheses. The most insightful finding is vocabulary size, which is much larger for the kicker reports. This size difference causes a cascade of under-performances in respect to style-similarity. Stylistic differences can be found which makes our output easily distinguishable from the original kicker reports. We conclude that given our approach, we are not capable of producing sufficiently imitating reports.

K Lessons Learned about Software Group Projects

One of the lessons we learned in software group projects is that even though we had a scrum-like workflow¹⁵, we often had issues coordinating our time. For example, some work depended on the finishing of another task by another person and if they ran into issues finishing on time, everything got delayed. In these situations, we learned that it is important to prepare as much as possible beforehand, so that work can continue immediately once the missing results are available. With five people working together, it is also quite likely that someone will be sick or busy at some point. Therefore, documenting code and ideas properly is also important so that others can understand the work and take over if needed. We also learned that it’s important to keep each other updated on the current status of things and, along with that, regularly merge branches to decrease the risk of someone accidentally working with or building upon old code.

L AI Assistance

AI tools were used to support several implementation and formatting tasks throughout the project. The manual annotation tools, including `annotate.py` and the visualization tool, were developed with AI assistance. AI was also used to rewrite the rule-based approach so that its output format was consistent with the other Content Selection methods. Further assistance included debugging and sanity checks, adding logging to the BiLSTM training scripts, fixing import and path issues after restructuring the repository, and formatting the results of the error analysis. In addition, AI was used for $\text{\LaTeX}$ table formatting, writing code for plots and proofreading.

¹⁵<https://www.scrum.org/resources/scrum-guide>