DeformAR: A Visual Analytics Framework for Evaluation of Arabic Named Entity Recognition

Anonymous ACL submission

Abstract

Arabic Named Entity Recognition (ANER) presents challenges due to its linguistic characteristics (Qu et al., 2023). We present DeformAR, a visual analytics framework for evaluating and interpreting Arabic NER models via a structured, component-based approach. DeformAR combines quantitative metrics and qualitative visualisation across data and model subcomponents to identify performance weaknesses and explain system behaviour. In a case study on ANERCorp, DeformAR identifies annotation mistakes, explains calibration issues, and reveals interaction effects between subcomponents. To our knowledge, this is the first framework to support both evaluation and interpretability for Arabic NER.

1 Introduction

Arabic Named Entity Recognition (ANER) presents a unique challenge in the field of NLP (Darwish et al., 2021). Arabic is a morphologically rich language, lacks standardised tokenisation, and has orthographic variations, all of which complicate NER (Shaalan, 2014). Despite recent improvements introduced by transformer-based models (Devlin et al., 2019; Antoun et al., 2020; Patwardhan et al., 2023), ANER remains under-explored, particularly in how we evaluate and understand model performance. Existing evaluation techniques in NER, particularly in Arabic (Fu et al., 2020; Obeid et al., 2020) focus on quantitative analysis and aggregate metrics such as F1 scores, which offer limited insight into model behaviour and performance understanding. While more detailed evaluation and interpretability tools have focused on English and text classification, NER, and especially ANER, has been underexplored (Sun et al., 2021; Ruder et al., 2022).

To address this gap, we present DeformAR, a visual analytics framework designed for evaluating and analysing Arabic NER systems. DeformAR adopts a structured, component-based framework that integrates quantitative metrics and qualitative visualisation, by dividing the task into two main components: the model and the data. Each component is further divided into subcomponents that interact during fine-tuning. For example, the data component includes subcomponents such as the vocabulary and NER annotations, while the model component includes the embedding representations and output layer. Each subcomponent exhibits distinct behaviours and also interacts with others in ways that affect overall performance. DeformAR supports the analysis of these behaviours and interactions using both quantitative metrics and qualitative visualisations.

In the quantitative stage, DeformAR measures the behaviour and properties of each subcomponent in isolation, offering insight into where weaknesses are located. These findings inform the qualitative phase, which combines interactive visualisations with interpretability techniques. By linking token-level behavioural metrics to visual analytics, users can identify patterns, explain them using quantifiable metrics, and explore specific examples for supporting evidence. Together, these stages help answer three key questions: what the weaknesses are, how they occur, and why.

This work makes three main contributions. First, we propose a component-based methodology for evaluating both model and data subcomponents, as well as their interactions. Second, we introduce an interactive dashboard that integrates quantitative and qualitative analysis. Third, we demonstrate how DeformAR can identify and explain patterns, supporting more detailed evaluation of Arabic NER systems. To our knowledge, DeformAR is the first framework to combine interpretability and evaluation for Arabic NER, addressing a gap in both the NER and Arabic NLP literature.

The rest of this paper is organised as follows. Section 2 describes the architecture of DeformAR,

including its extraction pipeline, subcomponent metrics, and dashboard design. Section 3 presents a case study on ANERCorp, combining quantitative and qualitative analysis. Section 4 situates our work within the literature. Section 5 discusses key findings and future directions, followed by conclusions in Section 6.

2 Framework Architecture

DeformAR consists of two main phases: an extraction phase and a dashboard phase. In the extraction phase, we identify subcomponents within both the model and the data, and capture their interactions during fine-tuning. In this analysis, we determine what can be extracted at inference time to explain system weaknesses. The dashboard phase then presents this information through quantitative metrics and interactive visualisations. Below, we describe each phase in detail.

2.1 Data Extraction Phase

The first step in the extraction phase is data preparation, which supports various types of NER datasets and unifies them into a standard format for preprocessing. The preprocessing involves aligning words with their true labels, applying tokenisation to words, and converting both tokens and labels into numeric representations. Once processed, the data is used for model fine-tuning.

During fine-tuning, the model and data subcomponents begin to interact, as illustrated in Figure 1. The raw data consists of word sequences annotated with named entity tags, which serve as the true labels. On the model side, the key subcomponents include the representation module, output layer, and loss function. Before fine-tuning, words are tokenised using a WordPiece tokenizer. Following standard NER practice, only the first subword of each tokenised word is assigned the entity tag (Antoun et al., 2020; Devlin et al., 2019), these are referred to as core tokens. As a result, the tokenised vocabulary becomes a mix of original words and the first subwords of the tokenised ones.

Although only the first subword of each tokenised word (the core token) is aligned with the true label, the representation subcomponent processes the full subword sequence. The resulting embeddings are passed to a multi-layer perceptron (MLP), which generates logits for each token. A cross-entropy loss is then computed, but only for core tokens. As a result, while all tokens contribute

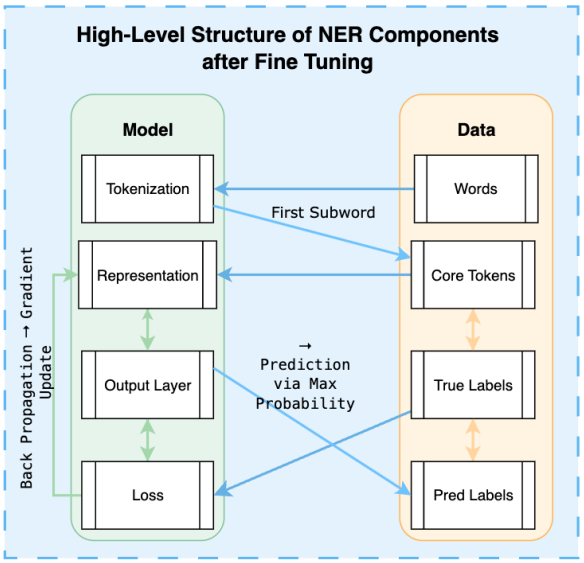

Figure 1: Overview of the interaction between model and data subcomponents during fine-tuning. **Orange** arrows represent interactions within data subcomponents, **Green** arrows represent interactions within model subcomponents, and **Blue** arrows represent cross-component interactions between model and data.

to contextualisation and logits, only the core tokens influence parameter updates via backpropagation. This highlights a key distinction: non-core tokens are involved in forward computation but ignored during loss computation and optimisation.

This process reveals several subcomponents within both the model and the data, each has its own characteristics and interaction patterns, either within a component (e.g., core tokens and true labels) or across components (e.g., words and the tokenizer). In the next section, we describe the extracted data and the metrics used to characterise these subcomponents and their interactions.

2.2 Subcomponent Metrics

This section outlines the metrics used to characterise system subcomponents, capturing both individual behaviour and interactions. For the data component, we focus on two subcomponents: core tokens and true labels. We compute several metrics to describe each individually, as well as their interaction. Structural metrics include dataset size and the distribution of tags, both at the entity tag level (e.g., B-LOC, I-PER) and at the span level.

Lexical characteristics are captured using metrics such as lexical diversity, defined as the ratio of unique token types to total tokens. "Tokens" may refer to either original words or core tokens (i.e., first subwords), allowing us to distinguish pre-

and post-tokenisation vocabularies. Another key metric is `entity tag overlap`, which measures the number of token types associated with multiple entity tags. We also calculate a tag-specific out-of-vocabulary (OOV) rate, defined as the number of unique token types associated with a given entity tag in the test set that do not appear with the same tag in the training set, divided by the total number of unique token types for that tag in the test set.

In addition to structural and lexical metrics, we compute behavioural metrics that capture interactions between subcomponents. The first is the tokenisation rate, which measures the average number of subwords generated per word. To characterise the relationship between core tokens and true labels, we use label inconsistency, which measures how often a token appears with different entity tags in the training data. For instance, if the token "university" (labelled as B-LOC in the test set) appears five times in the training data, three times as B-LOC and twice as O, its inconsistency ratio is 2/5. We also compute ambiguity using Shannon entropy, which measures the uncertainty in a token's label distribution. For each test token, we calculate its label distribution in the training data and calculate entropy; higher scores indicate more annotation ambiguity. Tokens not observed in training are assigned a default value of –1.

For the model, we extract token-level loss values and output probabilities. From the probabilities, we derive prediction confidence, the model's certainty in its top prediction (i.e., max probability) and prediction uncertainty, which reflects how evenly probabilities are spread across all labels (entropy). To assess the representation layer, we compute silhouette scores[1] to measure tag separation in embedding space, and apply UMAP for 2D projection and K-means clustering to assess alignment with gold labels.

These metrics vary in granularity: some (e.g., lexical diversity) are corpus-level, while others (e.g., loss or confidence) are token-level. In the quantitative phase, we examine them in isolation; in the qualitative phase, we explore their relationships through interactive visualisations. The next section describes how these metrics are presented in the dashboard.

[1]Using `silhouette_samples` from scikit-learn

2.3 Dashboard Phase

The dashboard presents the extracted outputs via an interactive interface built with Plotly Dash, enabling both quantitative and qualitative analysis. It consists of three main tabs: Quantitative Analysis, Qualitative Analysis, and Instance-Level View. Each tab provides different visualisations and supports user interaction through components such as dropdowns, buttons and selection tools.

2.4 Quantitive Analysis Tab

The Quantitative Analysis Tab (Figure 2) provides a high-level, metric-driven overview of the NER system. It is divided into three sections: **Evaluation Metrics**, **Data Component**, and **Model Component**. Each section focuses on analysing metrics in isolation, without considering relationships between them. This analysis helps quantify the behaviour and performance of subcomponents, providing insights into system strengths and weaknesses.

The **Evaluation Metrics** section (red squares) examines the alignment between true and predicted labels using standard metrics and confusion analysis. The **Data Component** section (yellow squares) presents structural, lexical, and behavioural metrics that characterise the dataset. The **Model Component** section (green squares) includes metrics that describe prediction behaviour and representation quality. The **Tokenisation Rate** (blue) is a cross-component metric capturing interaction between model and data. The tab supports user interaction through dropdown menus and buttons, with visualisations mainly consisting of bar charts and heatmaps aggregated by entity tag.

2.5 Qualitative Analysis Tab

The Qualitative Analysis Tab (Figure 3) focuses on exploring relationships between subcomponents through interactive visualisations and interpretability techniques. Unlike the Quantitative Tab, it allows users to investigate multiple variables simultaneously, identify patterns in the representation space, and interpret these patterns using behavioural metrics and categorical variables.

The tab supports two types of interaction: standard controls (e.g., dropdown menus) and cross-linked interactivity, where actions in one view update others (as shown by the dashed arrows in Figure 3). The **Filtering Section** allows users to filter tokens using two mechanisms: dropdowns

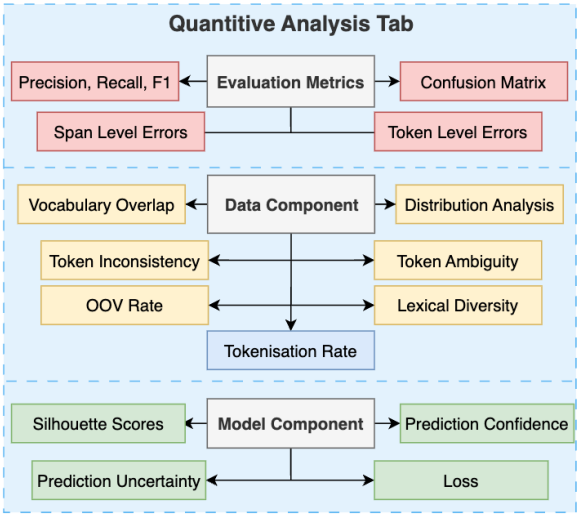

Figure 2: Overview of the Quantitative Analysis Tab in DeformAR.

based on categorical variables (e.g., true labels, error types), and a table-based filter driven by behavioural metrics. For more details, see Appendix A.1.

The **Filtering Section** affects the **Behavioural Analysis Section**, the core section of the tab. This section includes three linked views: a **2D UMAP projection** of token embeddings, a **Behavioural Metric Scatter Plot**, and a **Heatmap** showing pairwise correlations between behavioural metrics (Pearson or Spearman). Selecting a cell in the heatmap sets the x- and y-axes of the scatter plot (see Appendix A.2), while the scatter plot and UMAP are bidirectionally linked, selecting points in one highlights them in the other. To our knowledge, this level of interactivity is unique to DeformAR, allowing users to explore up to six variables simultaneously. For example, users can colour the UMAP by true labels and confusion matrix outcomes (e.g., FP or FN), and examine the same points in the metric scatter by loss and confidence grouped by error types. This enables detailed analysis of whether spatial regions in the embedding space align with error patterns and how they are reflected in model metrics. Another auxiliary section is the **Selection Summary**, which updates whenever a selection is made in either the behavioural or UMAP scatter plot. It provides summary statistics for the selected tokens across all metrics. For further details, see Appendix A.3.

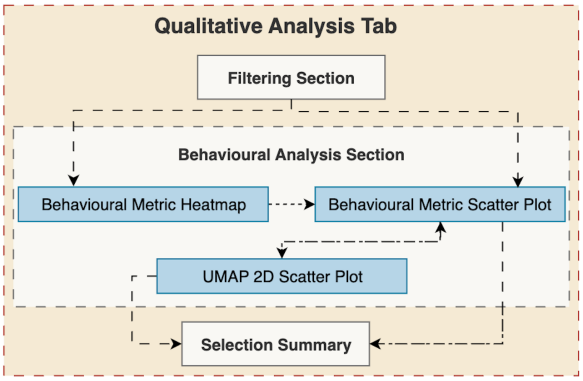

Figure 3: Overview of the Qualitative Analysis Tab in DeformAR.

2.6 Instance Level Tab

The Instance-Level Tab (Figure 4) supports fine-grained analysis of specific tokens and their surrounding context. It is divided into three sections: **Sentence Viewer**, **Token Analysis**, and **Attention Analysis**. This tab is linked to the Qualitative Analysis Tab: when users select tokens in the UMAP or scatter plots, only sentences that include those tokens can be viewed in the sentence viewer.

The **Token Analysis** section supports exploration of individual tokens. Users can view the distribution of entity tags assigned to a selected token across training and test splits, and compute cosine similarity to identify the most similar tokens in the dataset. This similarity-based approach aligns with example-based interpretability methods, helping identify influential instances that may have contributed to the model's prediction. Users can also examine these tokens within their original sentence context using the **Token Context View**. For more details on token similarity and filtering, see Appendix A.4.

The **Attention Analysis** section allows users to inspect the model's attention behaviour for a selected sentence. We integrate BertViz (Vig, 2019) to visualise token-level attention patterns. We also compute attention similarity between the pre-trained and fine-tuned models, presented as a heatmap, to highlight changes introduced by fine-tuning. This helps identify where and how attention shifts occur across layers or heads.

3 Arabic NER Case Study

In this section, we demonstrate the capabilities of DeformAR using the ANERcorp dataset, a standard Arabic NER corpus introduced by Benajiba et al. (2007) and standardised by CAMeL Lab

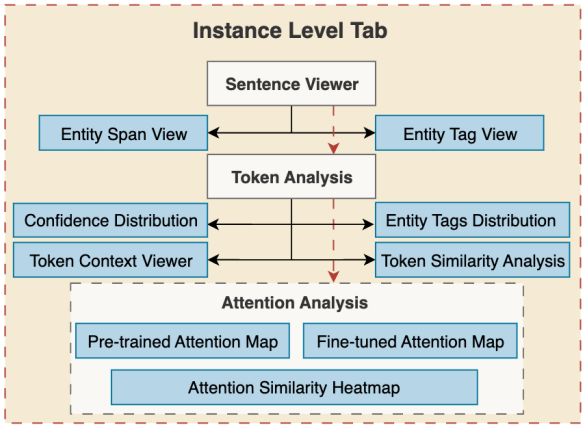

Figure 4: Overview of the Instance-Level Tab in DeformAR.

Entity Type	Precision	Recall
LOC	0.893	0.934
MISC	0.772	0.634
ORG	0.784	0.751
PER	0.860	0.844

Table 1: Precision and recall by entity type under the IOB2 tagging scheme.

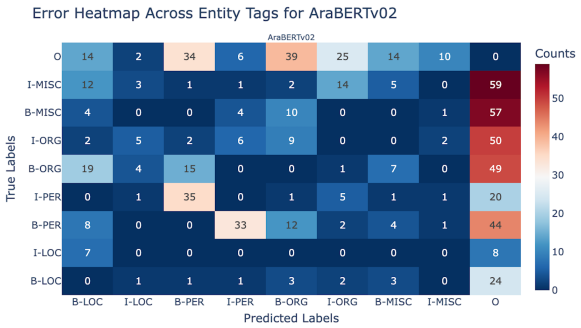

Figure 5: Confusion heatmap showing predicted versus true entity tags.

(Obeid et al., 2020). We use the CAMeL Lab version, which includes minor corrections, and provides a sequential train/test split (5/6 training, 1/6 testing) based on word count, resulting in approximately 125K and 25K words respectively. The dataset consists of Modern Standard Arabic text annotated using the IOB2 scheme across four entity types: PER, ORG, LOC, and MISC.

We fine-tune `AraBERTv02-base`, a 12-layer transformer with 768 hidden units and 12 attention heads. A linear classification layer is added on top of the final hidden state to produce token-level logits. The model is trained with a cross-entropy loss function.

Fine-tuning is performed using the AdamW optimiser with a learning rate of 5e–5, a batch size of 16, and four training epochs. Evaluation is performed using the seqeval library (Nakayama, 2018) in strict mode under the IOB2 tagging scheme. Additional training and evaluation details are provided in Appendix A.5.

3.1 Quantitative Analysis

In this section, we present cross-component findings derived from both model and data subcomponents. We begin by examining overall performance trends across entity spans, followed by an analysis of data characteristics that may help explain these patterns.

As shown in Table 1, precision exceeds recall across all entity spans except for `LOC`, where recall is slightly higher. The model demonstrates relatively high precision overall, indicating a tendency to minimise false positives. The most significant performance drop is seen in `MISC`, which has both the lowest precision (0.772) and recall (0.634), followed by `ORG`. In contrast, lower recall indicates that true spans are often missed.

To further understand the performance gap across entity spans (e.g., `LOC`), we examine prediction errors at the level of individual entity tags (e.g., `B-LOC`, `I-LOC`). Figure 5 presents a confusion heatmap showing the distribution of true versus predicted entity tags. A dominant pattern is the misclassification of entity tag tokens as `O`, particularly for lower-performing spans such as `MISC` and `ORG`. We refer to these misclassifications as exclusion errors. In addition to exclusion errors, two other patterns emerge: boundary errors, where the model confuses `B-PER` and `I-PER`, and inclusion errors, where `O` tokens are mistakenly predicted as entities, particularly for `B-ORG` and `B-PER`. These error patterns help explain the observed gap between precision and recall: `MISC` spans often suffer from exclusion errors that reduce recall, while `LOC` spans, which show fewer errors of that type, achieve higher recall.

Several data characteristics can potentially explain the observed performance trends. One key factor is tokenisation. As mentioned in Section 2, after tokenisation the vocabulary becomes a mix of words and first subwords. While this does not affect the overall structure or number of core tokens, it changes the lexical structure of the dataset. For example, the number of unique named entity words in the training split drops from 4,069 to 3,445 after

tokenisation (15.34% reduction). This has implications for the dataset's lexical characteristics, including increased overlap across entity tags. Further details are provided in Appendix A.6.

To better understand how data and model subcomponents contribute to performance variation, we analysed the metrics discussed in Section 2.2. Although our analysis is conducted per entity tags (e.g., `B-LOC`, `I-LOC`), for simplicity we refer to these collectively by their span type (e.g., `LOC`, `PER`) for the rest of this section. **On the data side**, we analysed the relationship between core tokens and true labels and identified three contributing factors: structural and lexical differences, span complexity, and annotation inconsistency. Tokens associated with low-performing spans such as `MISC` and `ORG` have fewer instances, higher lexical diversity, more OOV tokens, and higher tag overlap with `O`. In contrast, `LOC` tags appear more frequently, show lower diversity and OOV, and have minimal overlap with `O`. `PER` spans fall in between, with moderate OOV and diversity, and high overlap with `O`. In terms of **span complexity**, we found that the length of `LOC` spans is much smaller than others, especially `PER` and `ORG`, which are much longer. In all cases, train spans are longer than test spans, except for `MISC`, where the opposite is true. This means the model is exposed to shorter spans during training but is asked to predict longer spans at test time, which may contribute to poor generalisation. In terms of inconsistency, we found that `ORG` and `MISC` have much higher inconsistency compared to other entity tags. Further details and supporting evidence are provided in Appendix A.7.

On the model side, we observed several trends that align with the difficulties in data characteristics. First, lower-performing tags such as `MISC` and `ORG` show the highest token-level loss values, with `I-MISC` being the most prominent. When examining prediction confidence, we found that the model tends to assign high confidence regardless of whether the prediction is correct or incorrect. This pattern is especially notable in exclusion errors, where entity tokens are misclassified as `O`; in these cases, confidence remains high despite the error. This behaviour suggests calibration issues — the model's predicted probabilities do not align with actual correctness. To confirm this, we examined prediction uncertainty across the probability distribution. We found that the model tends to be less uncertain (i.e., more confident) when predicting correctly for tags like `LOC` and `PER`, but more uncertain when predicting `MISC` and `ORG`. This suggests that the model is often unsure whether its predictions are correct in the more challenging spans.

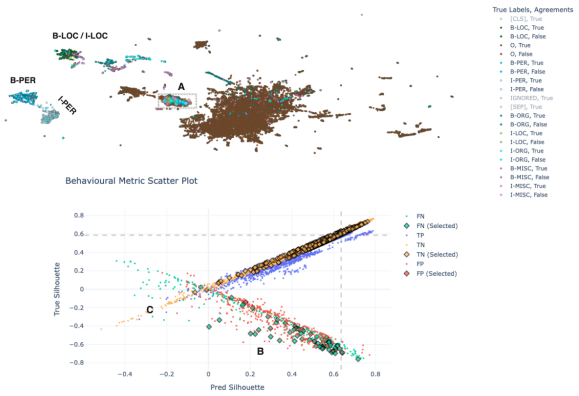

Figure 6: **Top**: UMAP projection of the final hidden states, coloured by true labels and prediction correctness. **Bottom**: Scatter plot of true vs. predicted silhouette scores, coloured by confusion matrix outcomes labels.

3.2 Qualitative Analysis

We begin the qualitative analysis by examining the representation space using a UMAP projection, shown in Figure 6. We observe three main patterns: well-separated regions, error patterns, and anomalies. **Well-separated regions** include distinct clusters for high-performing spans such as `PER` and `LOC`, where `B` and `I` tags form compact and clearly defined areas. In contrast, spans like `ORG` and `MISC` are more scattered and tend to overlap with the dominant `O` region. Finally, a large, dense region of `O` tokens dominates the space. **Error patterns** often appear within these dense regions. For instance, the `O` region includes tokens from other classes, such as `LOC`, indicating exclusion errors where entity tokens are misclassified as `O`. **Anomalies** are more subtle and involve correctly predicted tokens that appear far from their expected class region. For example, in the `LOC` region, there are `O` tokens that are correctly predicted despite being located in an unexpected area.

The behavioural scatter plot (bottom) shows how these patterns are reflected in silhouette-based metrics. Tokens with high agreement between true and predicted silhouette scores (top right) are typically true positives and true negatives (i.e., `O` tokens correctly predicted), while low-separation tokens (bottom right) are often false positives or false negatives. These metrics are not only useful for interpretation, but also for detecting anomalies. For example, the anomaly observed in the `LOC` region, where

O tokens are correctly predicted but located far from their expected region, is highlighted by the point C of the behavioural scatter plot. These tokens show low silhouette scores for both true and predicted labels. When inspecting them in the instance-level view, we found they often correspond to annotation mistakes, inconsistent labelling, or contextually ambiguous cases. The model predicts them as O because they are wrongly labelled that way in the training data, even though the surrounding context may suggest a different semantic meaning, hence placed in another region.

We also investigated another interesting pattern highlighted by Region A in the UMAP projection, marked with a selection box. This selection is reflected in the behavioural metric scatter plot, where the region contains a mix of true negatives and misclassified tokens. Examining the errors in Region B, we found they correspond to systematic issues where certain tokens appear abruptly at the start of a sentence without clear semantic relevance, similar to the errors found by AlDuwais et al. (2024). However, we provide further clarification by comparing the Benajiba and CAMeL Lab versions of the dataset using the token context view. This comparison revealed that some tokens originally belonged to different sentences (in Benajiba) but were placed at the beginning of unrelated ones in the test set (in CAMeL Lab). These errors likely results from preprocessing or sentence segmentation errors introduced during dataset standardisation, rather than annotation inconsistencies.

Unlike annotation mistakes, which often result from unclear guidelines, these systematic errors arise from consistent issues in dataset preparation. Interestingly, tokens affected by this issue exhibit different behaviour from other types of errors: they show high confidence, high loss, and very low (often negative) silhouette scores. As shown in Figure 7, the highlighted tokens demonstrate very low uncertainty despite poor representation separability, reflecting a mismatch between the model's output and its internal representation. In contrast, tokens associated with annotation mistakes tend to show higher uncertainty, indicating that the model is less confident. These patterns are visualised in the highlighted tokens with large diamond markers. We also identified other error patterns during the instance-level analysis that result from tokenisation-related issues, such as the removal of diacritics, which introduces ambiguity, and malformed subwords that further confuse the model. For further details, see Appendix A.8.

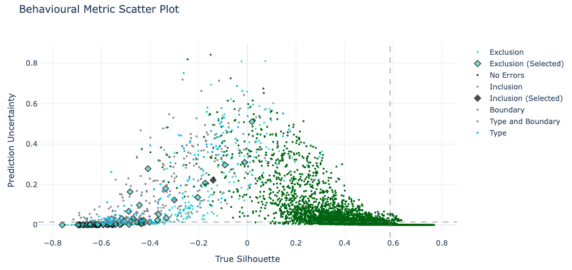

Figure 7: Behavioural Scatter Plot showing prediction uncertainty versus true silhouette score.

These observations point to a broader conclusion: there appears to be a misalignment between the learned representation space and the output layer, which in some cases results in conflicting decisions. This misalignment manifests differently depending on the error type, annotation-related errors tend to exhibit lower confidence and higher uncertainty, while systematic errors often produce high-confidence misclassifications despite weak representation separability. This helps explain the calibration issues noted in the quantitative analysis.

In terms of clustering alignment, we found that K-Means identified clusters that correspond to certain annotation patterns. For example, B-PER and I-PER were captured by separate, distinct clusters, while B-LOC and I-LOC were grouped into a single cluster. In contrast, spans such as MISC and ORG were distributed across multiple clusters, confirming their weak representation structure. Further details are provided in Appendix A.9.

4 Related Work

We situate DeformAR within three areas of related work: interpretability techniques, visual analytics tools, and NER-specific evaluation methods. Interpretability methods are often categorised as global or local (Zini and Awad, 2023; Ferrando et al., 2024). Global approaches, such as Aken et al. (2019), use dimensionality reduction to visualise hidden state structure, while local methods include attention visualisation and attribution techniques. However, the explainability of attention remains debated (Sun et al., 2021; Zhao et al., 2024), and attribution methods (e.g., LIME, SHAP) are harder to adapt to sequence labelling tasks (Ruder et al., 2022). Example-based techniques such as influence functions (Sun et al., 2021; Jain et al., 2022) are rarely explored outside English. Dataset Cartography (Swayamdipta et al., 2020) is conceptu-

ally similar to our behavioural metrics, but focuses on training dynamics, whereas DeformAR targets inference-time analysis and integrates interpretability directly into evaluation.

Several visual analytics tools support the inspection of Transformer models, including LIT (Tenney et al., 2020), InterpreT (Lal et al., 2021), and T3-Vis (Li et al., 2021). Among these, T3-Vis is the most closely related: it offers cartographic visualisations of training dynamics across epochs and integrates attention analysis, with support for filtering based on confidence and prediction consistency. However, DeformAR extends this design by covering a broader set of subcomponents through various metrics and by linking behavioural quantitative metrics to qualitative analysis through interactive visualisations. While T3-Vis is guided by user-centric design goals, DeformAR is structured around the task itself: it decomposes NER into model and data components and examines their interactions. Unlike previous tools, DeformAR is specifically designed for NER and supports Arabic, both of which remain under-represented in interpretability research.

Other work in NER-specific evaluation focuses on annotation errors. For example, CLEANANERCorp (AlDuwais et al., 2024) used semi-automated methods to revise ANERCorp and similar corrections applied to CoNLL-2003 (Liu and Ritter, 2023; Rücker and Akbik, 2023). While DeformAR is not designed for correction, it uncovered similar issues (e.g., sentence-start anomalies) with minimal manual effort, while offering more explainability of their causes and impact.

Finally, bucket-based evaluation has been used to analyse performance variation by partitioning data into interpretable slices, such as entity length or frequency (Fu et al., 2020; Liu et al., 2021). While useful, these methods focus primarily on dataset-side properties. In contrast, DeformAR links data, model, and prediction behaviours, enabling deeper cross-component evaluation.

5 Discussion and Future Work

This section summarises the main contributions and outlines directions for future work. In the quantitative analysis, we presented a cross-component evaluation that showed how data characteristics can help explain performance differences and error patterns, and how these are reflected in model behaviour. We introduced behavioural metrics for both data and model subcomponents, emphasising their interactions. Although each metric was examined in isolation, their combined interpretation helped uncover broader trends such as annotation ambiguity and exclusion errors.

The qualitative analysis expanded on these findings by explaining how specific error types and calibration issues relate to representational structure and confidence behaviour. Through interactive visualisations, we were able to isolate annotation issues, systematic errors, and tokenisation artifacts with minimal manual effort. The integration of behavioural metrics with interpretability techniques, through the dashboard's interactive design, offers a step forward in bridging quantitative and qualitative evaluation. This strengthens the explainability of system outputs by revealing not only where models fail, but also how and why.

As next steps, one direction is to compare DeformAR's findings with re-annotated corpora such as CLEANANERCorp, to assess whether revisions resolve earlier issues or introduce new ones. Another is to expand the behavioural metric to include attention mechanisms or training dynamics. Finally, we plan to apply DeformAR to additional languages and tasks, and to study how architectural factors such as tokenizer design, model scale, or pretraining data influence NER performance.

6 Conclusion

We introduced DeformAR, a novel framework for interpreting and evaluating Arabic NER systems through structured, cross-component analysis. By combining token-level behavioural metrics with interactive visual analytics, DeformAR enables users to discover errors, understand representation structure, and interpret model behaviour. Our case study on ANERCorp demonstrates its ability to uncover both annotation and performance issues with minimal manual effort. DeformAR bridges the gap between interpretability and evaluation tools for Arabic NER, offering a foundation for future evaluation across tasks and languages.

Limitations

While DeformAR provides a structured and flexible framework for evaluating Arabic NER, several limitations remain. First, although many of the metrics and components are language-agnostic, some aspects — particularly instance-level analysis — require language-specific understanding, which may

limit generalisability to low-resource languages without further adaptation. Second, the current implementation focuses mostly on inference-time behaviour and does not incorporate training dynamics, which could offer additional diagnostic insight. Third, DeformAR currently supports up to two languages in the quantitative analysis and a single language in the qualitative analysis; extending support to multi-model comparison would enhance scalability for benchmarking. Finally, while the framework can identify potential annotation errors, it does not include automated correction mechanisms or integration with annotation workflows.

Acknowledgments

We thank the anonymous reviewers for their helpful feedback. Further acknowledgments will be added after the review period. We used ChatGPT to assist with grammar, punctuation, and formatting, as well as for summarising some related work papers and aligning our summaries with theirs. Additionally, we used it to help generate code for tables, figures, and occasionally their captions.

References

Betty van Aken, Benjamin Winter, Alexander Löser, and Felix A. Gers. 2019. How Does BERT Answer Questions? A Layer-Wise Analysis of Transformer Representations. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 1823–1832. ArXiv:1909.04925 [cs].

Mashael AlDuwais, Hend Al-Khalifa, and Abdulmalik AlSalman. 2024. CLEANANERCorp: Identifying and Correcting Incorrect Labels in the ANERcorp Dataset. In *Proceedings of the 6th Workshop on Open-Source Arabic Corpora and Processing Tools (OSACT) with Shared Tasks on Arabic LLMs Hallucination and Dialect to MSA Machine Translation @ LREC-COLING 2024*, pages 13–19, Torino, Italia. ELRA and ICCL.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. AraBERT: Transformer-based Model for Arabic Language Understanding. In *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, pages 9–15, Marseille, France. European Language Resource Association.

Yassine Benajiba, Paolo Rosso, and José Miguel BenedíRuiz. 2007. ANERsys: An Arabic Named Entity Recognition System Based on Maximum Entropy. In *Computational Linguistics and Intelligent Text Processing*, pages 143–153, Berlin, Heidelberg. Springer.

Kareem Darwish, Nizar Habash, Mourad Abbas, Hend Al-Khalifa, Huseein T. Al-Natsheh, Houda Bouamor, Karim Bouzoubaa, Violetta Cavalli-Sforza, Samhaa R. El-Beltagy, Wassim El-Hajj, Mustafa Jarrar, and Hamdy Mubarak. 2021. A panoramic survey of natural language processing in the Arab world. *Commun. ACM*, 64(4):72–81.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Javier Ferrando, Gabriele Sarti, Arianna Bisazza, and Marta R. Costa-jussà. 2024. A Primer on the Inner Workings of Transformer-based Language Models. *arXiv preprint*. ArXiv:2405.00208 [cs].

Jinlan Fu, Pengfei Liu, and Graham Neubig. 2020. Interpretable Multi-dataset Evaluation for Named Entity Recognition. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6058–6069, Online. Association for Computational Linguistics.

Sarthak Jain, Varun Manjunatha, Byron Wallace, and Ani Nenkova. 2022. Influence Functions for Sequence Tagging Models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 824–839, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Vasudev Lal, Arden Ma, Estelle Aflalo, Phillip Howard, Ana Simoes, Daniel Korat, Oren Pereg, Gadi Singer, and Moshe Wasserblat. 2021. InterpreT: An Interactive Visualization Tool for Interpreting Transformers. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 135–142, Online. Association for Computational Linguistics.

Raymond Li, Wen Xiao, Lanjun Wang, Hyeju Jang, and Giuseppe Carenini. 2021. T3-Vis: visual analytic for Training and fine-Tuning Transformers in NLP. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 220–230, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Pengfei Liu, Jinlan Fu, Yang Xiao, Weizhe Yuan, Shuaichen Chang, Junqi Dai, Yixin Liu, Zihuiwen Ye, and Graham Neubig. 2021. ExplainaBoard: An Explainable Leaderboard for NLP. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 280–289, Online. Association for Computational Linguistics.

Shuheng Liu and Alan Ritter. 2023. Do CoNLL-2003 Named Entity Taggers Still Work Well in 2023? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8254–8271, Toronto, Canada. Association for Computational Linguistics.

Hiroki Nakayama. 2018. {{seqeval}: a python framework for sequence labeling evaluation}. {Software available from \url{https://github.com/chakki-works/seqeval}}.

Ossama Obeid, Nasser Zalmout, Salam Khalifa, Dima Taji, Mai Oudah, Bashar Alhafni, Go Inoue, Fadhl Eryani, Alexander Erdmann, and Nizar Habash. 2020. CAMeL Tools: An Open Source Python Toolkit for Arabic Natural Language Processing. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 7022–7032, Marseille, France. European Language Resources Association.

Narendra Patwardhan, Stefano Marrone, and Carlo Sansone. 2023. Transformers in the Real World: A Survey on NLP Applications. *Information*, 14(4):242. Number: 4 Publisher: Multidisciplinary Digital Publishing Institute.

Xiaoye Qu, Yingjie Gu, Qingrong Xia, Zechang Li, Zhefeng Wang, and Baoxing Huai. 2023. A Survey on Arabic Named Entity Recognition: Past, Recent Advances, and Future Trends. *arXiv preprint*. ArXiv:2302.03512 [cs].

Sebastian Ruder, Ivan Vulić, and Anders Søgaard. 2022. Square One Bias in NLP: Towards a Multi-Dimensional Exploration of the Research Manifold. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2340–2354, Dublin, Ireland. Association for Computational Linguistics.

Susanna Rücker and Alan Akbik. 2023. CleanCoNLL: A Nearly Noise-Free Named Entity Recognition Dataset. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8628–8645, Singapore. Association for Computational Linguistics.

Khaled Shaalan. 2014. A Survey of Arabic Named Entity Recognition and Classification. *Computational Linguistics*, 40(2):469–510. Place: Cambridge, MA Publisher: MIT Press.

Xiaofei Sun, Diyi Yang, Xiaoya Li, Tianwei Zhang, Yuxian Meng, Han Qiu, Guoyin Wang, E. Hovy, and Jiwei Li. 2021. Interpreting Deep Learning Models in Natural Language Processing: A Review. *ArXiv*.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A. Smith, and Yejin Choi. 2020. Dataset Cartography: Mapping and Diagnosing Datasets with Training Dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9275–9293, Online. Association for Computational Linguistics.

Ian Tenney, James Wexler, Jasmijn Bastings, Tolga Bolukbasi, Andy Coenen, Sebastian Gehrmann, Ellen Jiang, Mahima Pushkarna, Carey Radebaugh, Emily Reif, and Ann Yuan. 2020. The Language Interpretability Tool: Extensible, Interactive Visualizations and Analysis for NLP Models. *arXiv preprint*. ArXiv:2008.05122 [cs].

Jesse Vig. 2019. A Multiscale Visualization of Attention in the Transformer Model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 37–42, Florence, Italy. Association for Computational Linguistics.

Haiyan Zhao, Hanjie Chen, Fan Yang, Ninghao Liu, Huiqi Deng, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, and Mengnan Du. 2024. Explainability for Large Language Models: A Survey. *ACM Trans. Intell. Syst. Technol.*, 15(2):20:1–20:38.

Julia El Zini and Mariette Awad. 2023. On the Explainability of Natural Language Processing Deep Models. *ACM Computing Surveys*, 55(5):1–31. ArXiv:2210.06929 [cs].

Appendix

A.1 Filtering Section

The Filtering Section enables users to explore behavioural patterns by filtering tokens based on selected categorical variables and behavioural metrics. Each row in the table corresponds to a token and includes values for multiple metrics such as ambiguity, loss, confidence, and silhouette scores. Users can apply filters through dropdown menus or manually interact with each column, as shown in Figure 8

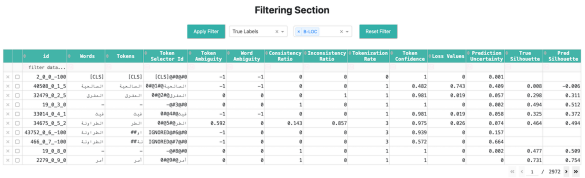

Figure 8: Screenshot of the Filtering Section used to inspect token-level metrics and apply dynamic filters.

A.2 Metric Interactivity

The behavioural metric interactivity links the heatmap and scatter plot in the Qualitative Analysis Tab. Clicking a cell in the heatmap selects a pair of metrics to plot against each other in the scatter view. The selected tokens can be coloured by two categorical variables one change the colour and the

other change the marker shape, and the plot is dynamically updated to support comparison across metrics.

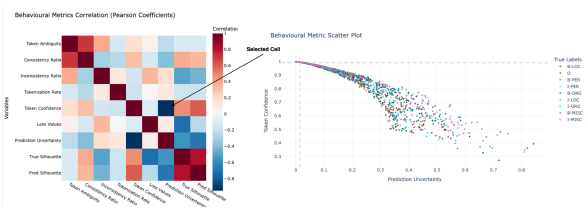

Figure 9: Interactive metric correlation and scatter plot. The selected cell controls the plotted axes.

A.3 Selection Summary

The Selection Summary view is triggered when tokens are selected in either the UMAP or behavioural metric scatter plots. On the left, users can choose a categorical variable (e.g., error type) for the x-axis, while the y-axis always represents the true label. On the right, two tables are provided: the categorical summary shows the distribution of the selected categorical variable, and the metric summary gives descriptive statistics for all behavioural metrics in the selected token group.

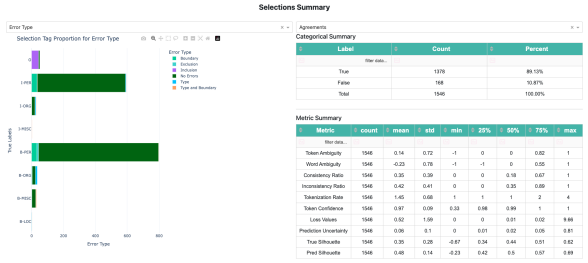

Figure 10: Selection Summary showing categorical and metric summaries for selected tokens.

A.4 Token Context and Origin Viewers

These views support sentence-level analysis by showing where a selected token appears in its context.

Figure 11 shows the Token Context View, which displays the sentence containing the selected token within either the training or test split of the CAMeL Lab version of ANERCorp.

Figure 11: Token Context Viewer showing the selected token within a sentence from the CAMeL Lab version.

Figure 12 shows the Token Origin View, which displays the sentence where the same token originally appeared in the Benajiba version of ANERCorp.

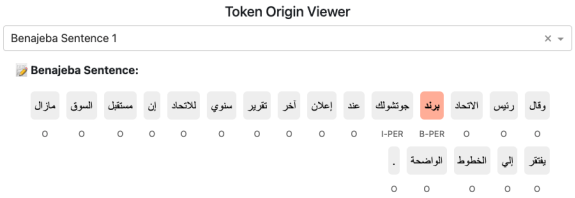

Figure 12: Token Origin Viewer displaying the sentence where the same token originally appeared in the Benajiba dataset.

A.5 Fine-tuning Hyperparameters

In addition to the optimiser and batch configuration described in Section 3, we apply the following hyperparameter settings during fine-tuning:

- Learning rate scheduler: linear decay with a warm-up ratio of 0.1
- Dropout: 0.1 (applied before the classification layer)
- Gradient clipping: maximum norm of 1.0
- Parameter freezing: all parameters are trainable except for LayerNorm and bias terms, which are frozen to improve training stability

Evaluation is conducted using the `seqeval` library (Nakayama, 2018), in strict (no-repair) mode under the IOB2 tagging scheme.

A.6 Tokenisation Impact

To illustrate the impact of tokenisation on lexical structure, we compare word-level and token-level type overlaps across entity tags in both train and test splits. As shown in Figure 13 and Figure 14, tokenisation increases the overlap across tags—particularly between entity tags and the O tag.

This increase in overlap may contribute to exclusion errors during inference, where entity tokens are misclassified as non-entities.

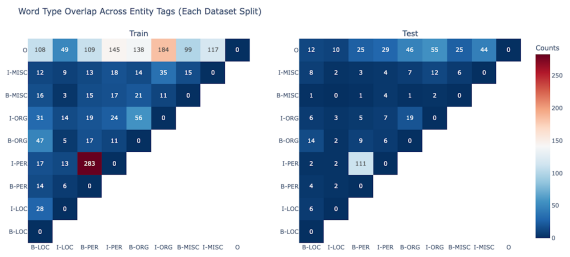

Figure 13: Word type overlap across entity tags (Train and Test splits).

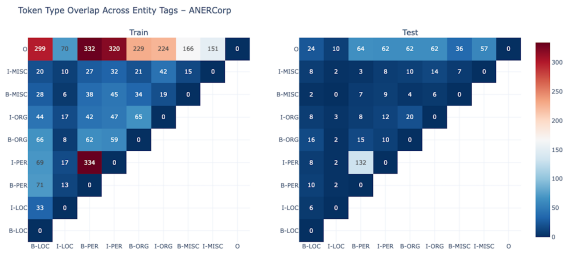

Figure 14: Token type overlap across entity tags (Train and Test splits).

Despite increased tag overlap, the effect of tokenisation on lexical diversity was minimal overall. As shown in Figures 15 and 16, most entity tags remain unaffected. The most notable change is seen in PER spans, where tokenisation reduced diversity more than for other spans.

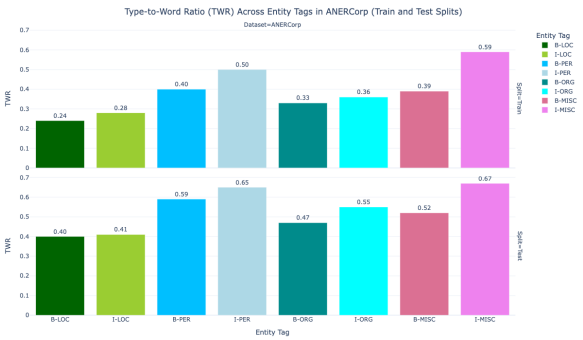

Figure 15: Type-to-word ratio (TWR) across entity tags in ANERcorp. Higher values indicate greater lexical diversity.

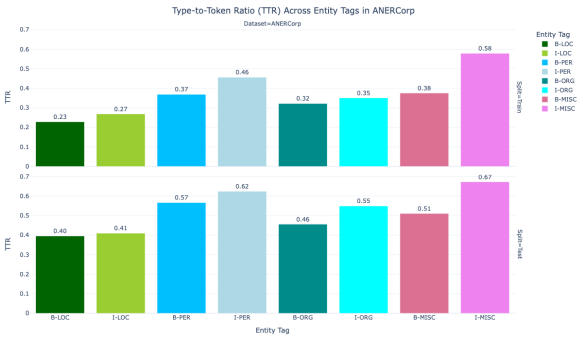

Figure 16: Type-to-token ratio (TTR) after tokenisation. The impact of tokenisation on diversity is limited.

A.7 Data Subcomponents

This section provides supporting evidence for the data-side analysis presented in Section 3, offering visual summaries for the metrics used to assess core token and label interactions, lexical diversity, annotation consistency, and span structure.

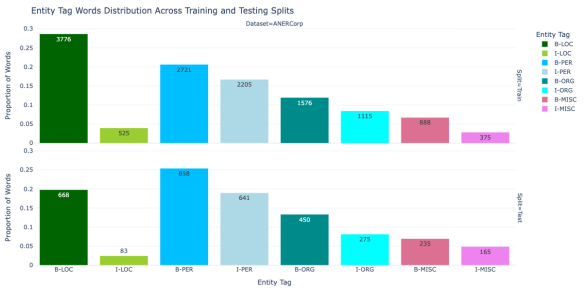

Figure 17: Distribution of entity tag across training and test splits.

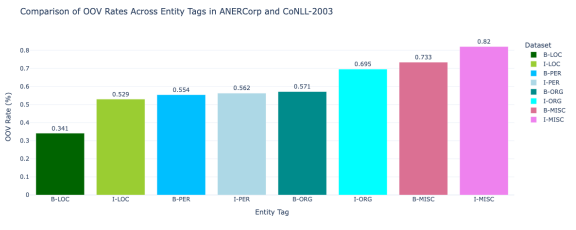

Figure 18: OOV rates by entity tag, showing the proportion of token types in the test set not seen with the same tag in training.

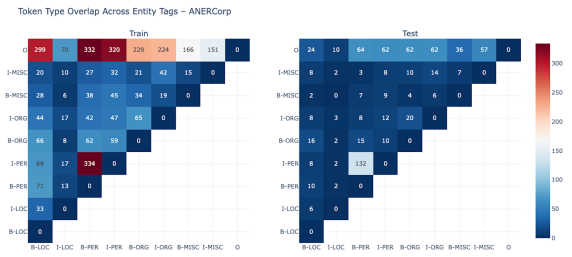

Figure 19: Entity tag overlap matrix showing the number of token types associated with multiple tags in training and test sets.

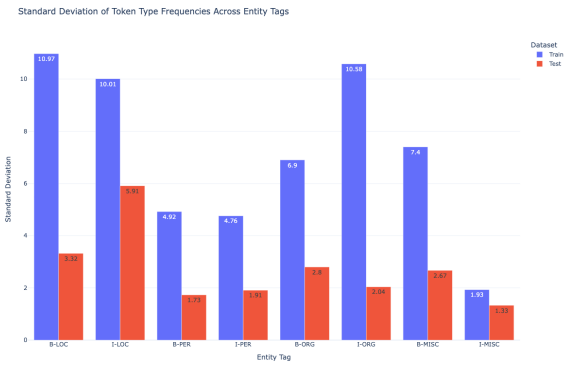

Figure 20: Standard deviation of token type frequencies across entity tags in training and test splits. For each entity tag, we compute how often each token type appears and calculate the standard deviation across those frequency counts. Higher values indicate skewed distributions with a few highly frequent types, while lower values suggest more uniform distributions.

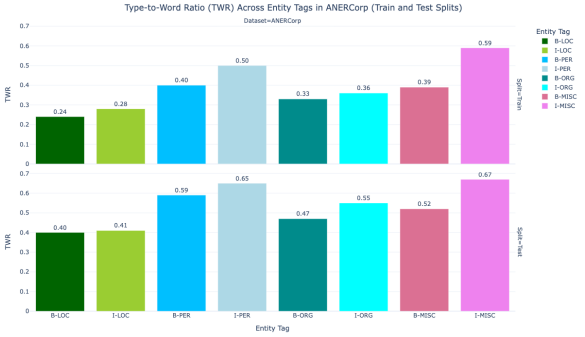

Figure 21: Lexical diversity (type-to-word ratio) across entity tags before tokenisation.

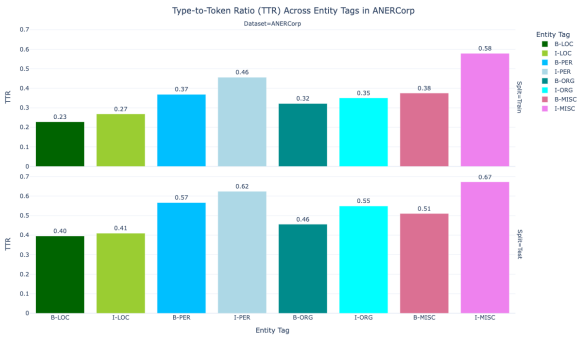

Figure 22: Lexical diversity (type-to-token ratio) across entity tags after tokenisation. The change due to tokenisation is minimal for most tags, with PER being the most affected.

Figure 23: Mean span length by entity type in training and test sets.

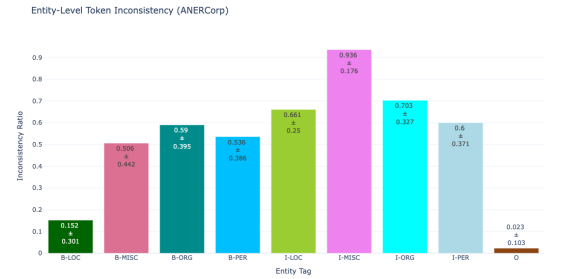

Figure 24: Token-level inconsistency ratio across entity tags. High values indicate that tokens are associated with multiple labels in the training data.

A.8 Examples from Various Error Patterns

This section presents examples from different error types observed during qualitative analysis. Each example highlights a specific source of error, sentence start, annotation inconsistency, or tokenisation ambiguity.

Sentence-Start Misalignment The top sentence shows a malformed example from the CAMeL ver-

sion where a proper name is placed abruptly at the start of the sentence, resulting in meaningless context. The bottom sentence shows the original Benajiba version, where the same token appeared mid-sentence in a more semantically meaningful context. This misalignment likely resulted from sentence segmentation errors during dataset standardisation and led to high-confidence misclassification.

Figure 25: Example of sentence-start misalignment between the CAMeL Lab version (top) and the original Benajiba sentence (bottom).

Diacritic Ambiguity This figure shows two words both spelled the same, however the first is Spanish and the second is Spain. The first word was mistakenly labelled as B-LOC potentially due to the absence of diacritics, while the second word is correctly labelled as B-LOC. When the model is exposed to this type of issue, the output layer predict the word as B-LOC due to the annotation patterns while the representation place the word according to its semantic meaning. Exposing the misalignments between the output layer and representation layer.

Figure 26: Inconsistent labelling due to diacritic ambiguity.

A.8.1 Tokenisation Ambiguity

Here, the token “Kat” was mispredicted as B-LOC despite it was referring to the word Catalonia. This is because the same first subword in Arabic is similar a country name called "Katanga". The model predicted it as `B-LOC`, likely due to similarity with training example where the token Kat was labelled as B-LOC. However, in the Catalonia sentence, the context does not support that label, so the model place it in the O region while predict it as B-LOC.

Figure 27: Ambiguity introduced by subword tokenisation. The token “Kat” was extracted from a longer place name.

A.9 Clustering Alignment

The `O` tag is split into three distinct clusters two dense regions corresponding to typical `O` contexts, and one smaller cluster associated with systematic errors such as sentence segmentation issues. This structure highlights the internal variability within the `O` class and supports the hypothesis that the model may overfit to this majority class.

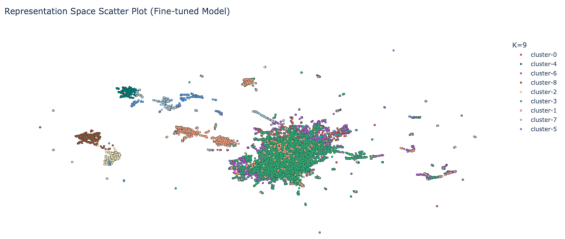

Figure 28: K-Means clustering of token representations in the fine-tuned embedding space ($k = 9$).

The other two clusters below cluster 3.

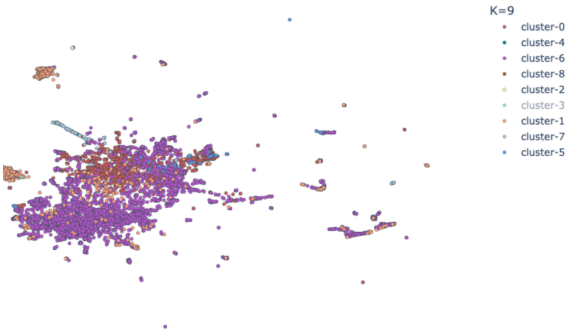

Figure 29: The two clusters assigned to the O tokens below cluster 3.