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

Ahmed Mustafa Younes
University of Sussex
Brighton, UK
ay227@sussex.ac.uk

Abstract

Arabic Named Entity Recognition (ANER) presents challenges due to its linguistic characteristics (Qu et al., 2023). While Transformer models have advanced ANER, evaluation still relies heavily on aggregate metrics like F1 score that obscure the interplay between data characteristics, model behaviour, and error patterns. We present DeformAR, a diagnostic visual analytics framework for evaluating and diagnosing Arabic NER systems through structured, component-level analysis and interpretability. DeformAR integrates quantitative metrics with interactive visualizations to support systematic error analysis, dataset and model debugging. In a case study on ANERCorp, DeformAR identifies annotation mistakes, model calibration issues, and subcomponent interaction effects. To our knowledge, this is the first open-source framework for component-level diagnostic evaluation and interpretability in Arabic NER, available at https://github.com/ay94/DeformAR.

1 Introduction

Arabic Named Entity Recognition (ANER) presents many challenges due to the linguistic properties of Arabic, including rich morphology, orthographic variation, and the lack of standardised tokenisation (Shaalan, 2014; Darwish et al., 2021). Although recent Transformer-based models have significantly improved ANER performance (Devlin et al., 2019; Antoun et al., 2020; Patwardhan et al., 2023), our ability to evaluate and interpret these systems has not progressed at the same pace.

Current NER evaluation practices rely primarily on aggregate metrics such as precision, recall, and F1 score. While useful for benchmarking, these metrics obscure how data properties, model representations, and prediction behaviour interact to produce errors (Fu et al., 2020; Obeid et al., 2020). In Arabic in particular, where annotation ambiguity, tokenisation effects, and lexical sparsity are common, aggregate scores provide limited guidance for diagnosing performance failures. More advanced interpretability and evaluation tools have largely focused on English and text classification tasks, leaving sequence labelling—and ANER in particular—underexplored (Sun et al., 2021; Ruder et al., 2022).

To address this gap, we introduce DeformAR, a diagnostic visual analytics framework for evaluating Arabic NER through structured, component-based analysis. Rather than treating NER system as a black box, DeformAR decomposes the system into interacting components—data (vocabulary, annotations), model (representations, output layer), and evaluation outputs—and analyses their interactions. Each subcomponent is characterised using multi-resolution metrics: from dataset-level statistics (tag distributions, lexical diversity) to token-level behavioural metrics (annotation ambiguity, prediction confidence, representation separability). This enables systematic diagnosis of where errors originate and why.

DeformAR operates in two phases: an extraction phase that builds the pipeline, fine-tunes models, and captures metrics; and a dashboard phase that links metrics through interactive visualisations.

Contributions This paper makes the following contributions:

- We propose a **component-based diagnostic evaluation methodology** for Arabic NER that decomposes systems into data, representation, and evaluation subcomponents and analyses their interactions.
- We introduce **DeformAR**, a visual analytics framework that integrates multi-resolution behavioural metrics with interactive visualisations, including a novel span-level error categorisation by extending `seqeval`.

Proceedings of the 2nd Workshop on NLP for Languages Using Arabic Script (AbjadNLP 2026), pages 253–275

- Through a detailed case study on ANERCorp, we demonstrate how DeformAR uncovers **hidden failure modes**—including annotation inconsistencies, tokenisation-induced ambiguity, and calibration failures—that are invisible to aggregate metrics.

The remainder of this paper is organised as follows. Section 2 describes the design of DeformAR, outlining the extraction and dashboard phases that correspond to our first two contributions. Section 3 presents a diagnostic walk-through on ANERCorp, demonstrating how DeformAR uncovers failure modes invisible to aggregate metrics, corresponding to our third contribution. Related work is discussed in Section 4, after the case study. This allows readers to first see DeformAR's diagnostic capabilities in action before contextualising them within the existing literature, making the distinctions from general-purpose tools (LIT, InterpreT) and training-focused systems (T3-Vis) more concrete. Finally, Sections 5 and 6 present discussion and conclusions.

2 Framework Design

Section 2.1 and Section 2.2 describe the two phases of DeformAR, corresponding to our first two contributions. We first outline the extraction phase, which captures component-level metrics across data, model, and evaluation subcomponents. We then describe the dashboard phase, which links these metrics through interactive visualisations to support hierarchical drill-down analysis.

2.1 Extraction Phase

The extraction phase comprises system configuration, fine-tuning, and metric extraction.

System Configuration DeformAR supports configurable NER pipelines, including model architecture, tokenisation strategy, output layer, and training setup. While the framework generalises to Transformer-based encoders and other sequence labelling tasks, our case study focuses on a single configuration (AraBERTv02 with a linear classifier), **prioritising depth over breadth to enable detailed analysis of component interactions** without the effects of multi-model shallow comparison.

Fine-Tuning During fine-tuning, model and data subcomponents interact in multiple ways, as illustrated in Figure 1. A key interaction involves tokenisation: words are tokenised using WordPiece,

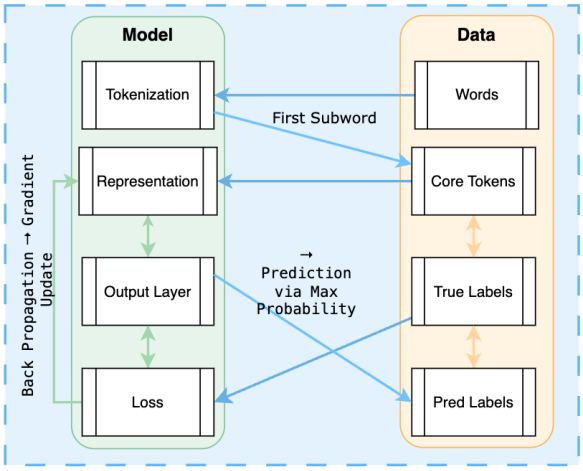

Figure 1: Overview of the interaction between model and data subcomponents during fine-tuning. **Orange** arrows represent interactions within data subcomponents, **Green** arrows represent interactions within model subcomponents, and **Blue** arrows represent cross-component interactions between model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

and only the first subword of each word is assigned an entity label (core tokens). While all subwords contribute to contextual representations, only core tokens contribute to loss computation and parameter updates. This asymmetry—where the effective vocabulary becomes a mix of full words and first subwords—affects how tokenisation influences learning, inference, and downstream behaviour. Additional interactions and a detailed core token example are provided in Appendix A.1.

Metric Extraction Having established how model and data subcomponents interact during fine-tuning, we now describe the multi-resolution metrics DeformAR extracts to characterise system behaviour. Here, multi-resolution refers to analysing the system at multiple granularities, ranging from corpus- and span-level statistics to token-level behavioural metrics. At the data level, we measure lexical diversity, tag distributions, ambiguity, inconsistency, tag overlap, and out-of-vocabulary rates. At the model level, we compute token-level loss, prediction confidence, uncertainty, and representation separability using silhouette scores. Cross-component metrics quantify how tokenisation alters lexical structure by recomputing data metrics on core tokens rather than words.

Following fine-tuning, we compute standard precision, recall, and F1 scores, and perform error analysis at both flat (B/I) and span levels. Span-level error categorisation is implemented by extend-

ing the seqeval evaluation pipeline to distinguish boundary, exclusion, and type errors. Full metric definitions are provided in Appendix A.2.

2.2 Dashboard Phase

The dashboard presents extracted metrics through three interconnected views, each targeting a different analytical granularity: the Cross-Component View for system-level comparison, the Behavioural Analysis View for token-level pattern exploration, and the Instance-Level View for sentence-specific inspection. Together, these views support a hierarchical drill-down workflow, enabling users to move from aggregate system-level patterns to token-level behaviour and finally to individual sentence-level instances, providing both global and local interpretability.

Cross-Component View The Cross-Component View supports metric-based comparison across data, model, and evaluation components using the metrics described in Section 2.1. As shown in Figure 2, the view is divided into three components where each component is rendered on a dedicated canvas with configurable analysis and visualisation options. Metrics can be compared per split or side-by-side using bar charts, heatmaps, and tables. Further interface and configuration details are provided in Appendix A.3.

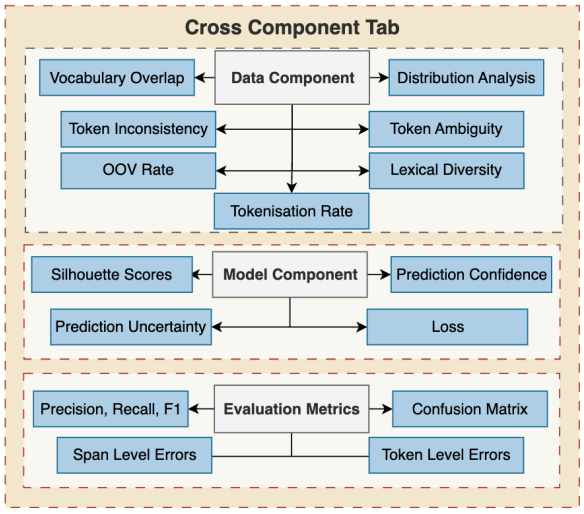

Figure 2: Overview of the Cross Component Tab.

Behavioural Analysis View The Behavioural Analysis View supports exploration of token-level metrics and representation structure through three linked visualisations: a Metric Correlation Heatmap, a Behavioural Scatter Plot, and a UMAP Projection Scatter Plot (Figure 3). The correlation heatmap displays pairwise relationships between behavioural metrics (e.g., ambiguity, confidence, loss, uncertainty). Selecting a cell in the heatmap dynamically assigns the corresponding metric pair to the axes of the Behavioural Scatter Plot, enabling targeted inspection of their interaction at the token level. The UMAP projection visualises the two-dimensional structure of contextual token embeddings after dimensionality reduction, allowing users to inspect representation organisation and overlap between entity types. Both scatter plots represent individual tokens as points, with configurable visual encodings (e.g., colour and shape) mapped to categorical variables such as ground-truth labels or error types.

Interactive linking across these views—dynamic axis updates, brushing[1], bidirectional synchronisation, and coordinated filtering—constitutes a key technical contribution of DeformAR. This design enables users to jointly explore relationships across up to six variables spanning data properties, representation quality, and prediction behaviour. For example, users can examine whether high annotation ambiguity correlates with low representation separability and high loss, and then verify whether such tokens form clusters in embedding space or correspond to specific error types. A filtering panel and selection summary further support slicing the data by categorical or numerical attributes and inspecting aggregate statistics for any selected subset. Additional implementation details are provided in Appendix A.4.

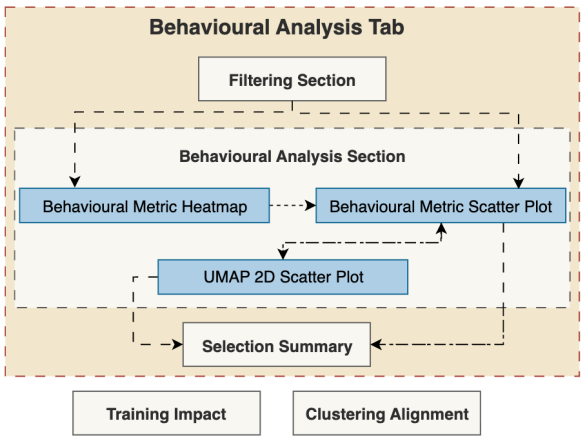

Figure 3: Overview of the Behavioural Analysis Tab in DeformAR. Dotted lines indicate interactive linking.

[1]Brushing refers to interactively selecting a subset of points (e.g., via rectangular or lasso selection) in one view, which highlights the corresponding points in linked views.

Metric	Train	Test
Total Words	125,102	25,008
Unique Words	29,252	9,075
NE Words	13,181	3,375
Unique NE Words	4,069	1,603

Table 1: ANERCorp dataset statistics.

Entity	Precision	Recall	# Examples
LOC	0.893	**0.934**	668
MISC	0.772	0.634	235
ORG	0.784	0.751	450
PER	0.860	0.844	858

Table 2: Performance and test-set number of examples by entity type.

Instance-Level View The Instance-Level View supports example-based inspection at the sentence and token levels, linking previously identified patterns to concrete instances (see Figure 14 in Appendix A.5). It comprises three modules.

The **Sentence Viewer** displays predicted and gold annotations using both span-level (e.g., LOC) and flat token-level (e.g., B/I-LOC) representations, supporting IOB1 and IOB2 schemes. Errors are highlighted directly within sentences to facilitate qualitative error analysis. The **Token Analysis Module** presents behavioural information for selected tokens, including prediction confidence, label probability distributions, and contextual similarity to other occurrences in the dataset. This enables tracing token behaviour across contexts and identifying influential or ambiguous training examples. The **Attention Analysis Module** visualises attention patterns before and after fine-tuning using BERTviz (Vig, 2019). A similarity heatmap compares attention distributions across layers and heads, highlighting components most affected by task-specific fine-tuning. Further details are provided in Appendix A.5.

3 Using DeformAR: A Diagnostic Walk-through

We demonstrate DeformAR's diagnostic capabilities through a case study on ANERCorp, a standard Arabic NER corpus introduced by Benajiba et al. (2007) and standardised by CAMeL Lab (Obeid et al., 2020). The dataset contains approximately 150K words of Modern Standard Arabic text annotated across four entity types (PER, ORG, LOC, MISC) using IOB2, with a sequential 5:1 train-test split (Table 1). We fine-tune AraBERTv02-base (12 layers, 768 hidden units) with a linear classification head using AdamW (learning rate 5×10^{-5}, batch size 16) for four epochs. Additional details are in Appendix A.6.

3.1 Diagnostic Questions

After fine-tuning, aggregate metrics reveal two patterns: a precision-recall disparity and substantial performance variation across entity spans (Table 2).

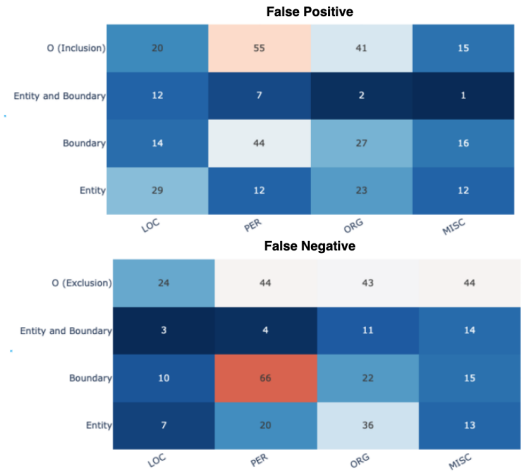

Figure 4: Span-level error type breakdown.

Performance falls into three groups: LOC and PER (high frequency, high performance), ORG (moderate), and MISC (low frequency, low performance).

To demonstrate DeformAR's diagnostic capabilities, we address four questions: **(1)** Why does a precision-recall gap emerge, and which spans contribute most? **(2)** Why does MISC perform substantially worse than LOC? **(3)** What properties distinguish high- from low-performing spans? **(4)** How does the core-token tokenisation strategy affect these outcomes?

The analysis proceeds view by view, focusing on salient insights. While not exhaustive, this demonstrates how DeformAR supports systematic pattern discovery, hypothesis generation, and explanation.

3.2 Stage 1: Cross-Component View

Why does a precision–recall gap exist? We begin by examining span-level error patterns—a novel capability enabled by extending the seqeval library beyond aggregate metrics to fine-grained span error categorisation. Figure 4 shows the distribution of false negative (FN) and false positive (FP) errors across entity spans.

For false negatives, exclusion errors (entity $\rightarrow$ O) dominate except for LOC, while boundary errors are prominent for PER. This explains why LOC exhibits higher recall than precision—it suffers fewer

exclusion errors—whereas other spans show the opposite. For false positives, boundary and inclusion errors dominate, with PER contributing the largest share. Flat token-level confusion patterns (Appendix A.7) reveal frequent B/I boundary mismatches and ORG-LOC confusions.

Overall, the precision–recall gap is driven primarily by high exclusion rates and boundary errors. LOC's strong recall reflects its relatively low exclusion rate compared to PER, MISC, and ORG.

Why does MISC underperform compared to LOC? While Figure 4 shows that MISC exhibits a higher error rate relative to its size—dominated by exclusion errors—this alone does not explain the performance disparity. The Cross-Component View reveals systematic differences in data properties and model behaviour.

Data characteristics: LOC is the most frequent span in training (3,776 examples), whereas MISC is the least frequent (888). However, frequency alone is insufficient to explain performance differences (e.g., I-LOC has fewer training examples than I-MISC yet achieves substantially higher F1). Lexically, LOC exhibits lower out-of-vocabulary rates and lower diversity, whereas MISC shows the highest values for both. Although both spans overlap substantially with the O tag, LOC's higher repetition enables clearer separation from O, while MISC's sparse and diverse examples hinder distinguishability.

Span structure further differentiates the two. LOC spans are shorter and simpler (fewer tokens per span), whereas MISC spans are longer and more complex. Token-type frequency analysis reveals that LOC has a concentrated distribution dominated by a small number of frequent types, while MISC exhibits a flatter distribution with many rare types, indicating higher lexical sparsity.

Annotation quality also differs. LOC shows ambiguity primarily in I-LOC tokens, with B-LOC relatively clean, whereas MISC exhibits high ambiguity and inconsistency across both B- and I-tags. Because LOC spans are simpler and dominated by beginning tokens, the impact of I-tag inconsistency is less severe than for MISC. Detailed statistics are provided in Appendix A.8.

Model behaviour: These data-level differences manifest in model behaviour. MISC exhibits the highest token-level loss and pronounced calibration issues: prediction uncertainty remains high even for correct predictions, particularly for I-MISC, and is similar for correct and incorrect B-MISC predictions. In contrast, LOC is well calibrated, showing low uncertainty when correct and high when wrong. Confidence distributions reveal that MISC exclusion errors (MISC $\rightarrow$ O) are often associated with high confidence, indicating confident misclassification. Silhouette scores reinforce this: I-MISC has negative values (poor separation), and B-MISC has very low scores, compared to consistently higher scores for B/I-LOC. Supporting evidence is in Appendix A.9.

Overall, MISC underperforms due to a compounding set of factors: low frequency, high lexical diversity, sparse token distributions, longer spans, and noisy annotations. These properties lead to weak representation structure, calibration failures, and high exclusion rates. In contrast, LOC benefits from abundant, repetitive, and cleaner data, resulting in well-separated representations, better calibration, and stronger performance.

Tokenization effects: Cross-component metrics reveal how tokenisation alters lexical structure. After WordPiece tokenisation, unique entity tokens decrease by 15.3% (from 4,069 to 3,445). While this slightly reduces OOV rates, it increases tag overlap as previously distinct words share first subwords, making disambiguation harder.

Through instance-level analysis, we identified how tokenisation introduces spurious ambiguity. For example, when distinct words with different entity labels share the same first subword, that core token receives conflicting supervision during training. For morphologically rich Arabic, this effect compounds existing challenges for MISC and ORG. Details and examples are in Appendix A.10.

3.3 Stage 2: Behavioural Analysis View

To examine whether performance differences identified in Stage 1 manifest in learned representations, we use the Behavioural Analysis View. This view enables discovery-driven exploration through interactive linking between behavioural metrics and visualizations, allowing users to jointly examine behavioural signals (confidence, uncertainty, loss), representation structure, and prediction outcomes.

Representation structure: Figure 5 (top) shows a UMAP projection of token embeddings. High-performing spans (LOC, PER) form compact, well-separated regions with stable B/I structure. In contrast, MISC and ORG exhibit scattered distributions overlapping heavily with the O region, mirroring the data-side characteristics identified earlier.

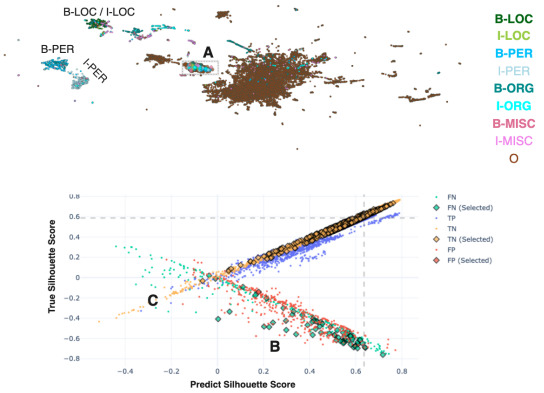

Figure 5: Top: UMAP projection of token embeddings coloured by entity label. Point shape encodes agreement between ground-truth and predicted labels (circle = correct, diamond = incorrect), as illustrated in Region A. Bottom: Behavioural analysis of predicted versus true silhouette scores. Tokens selected in Region A are highlighted using larger diamond symbols, with labels modified to indicate selection.

We quantify this using the linked behavioural scatter plot (Figure 5, bottom), which relates representation separability—measured via silhouette scores for true and predicted labels—to error types. Tokens with high separation under both labellings are predominantly correct, while those with low or negative separation correspond to false positives and negatives. This confirms that representation quality directly predicts error likelihood, linking embedding structure to output layer.

Span-specific behavioural patterns:

To examine whether performance differences reflect systematic differences in model behaviour, we compare correlations between confidence, uncertainty, and representation separability (silhouette scores) for LOC and MISC tokens.

For LOC, token confidence correlates positively with true silhouette score ($r = 0.65$): the model exhibits well-calibrated behaviour where high-confidence predictions correspond to well-separated representations. For MISC, this relationship is substantially weaker ($r = 0.28$), indicating misalignment between prediction confidence and representation quality. Detailed analysis reveals that MISC exhibits exclusion errors made with high confidence despite low representation separability, alongside correct predictions with low confidence and only moderate separation.

Figure 6 illustrates this misalignment through predicted silhouette versus uncertainty. While LOC shows clear separation—correct predictions achieve high silhouette and low uncertainty—MISC displays inverted patterns where some incorrect predictions achieve high predicted silhouette despite being wrong. These behaviours provide representation-level evidence for the calibration issues identified in Stage 1. Detailed correlation matrices and scatter plots for both spans are provided in Appendix A.11.

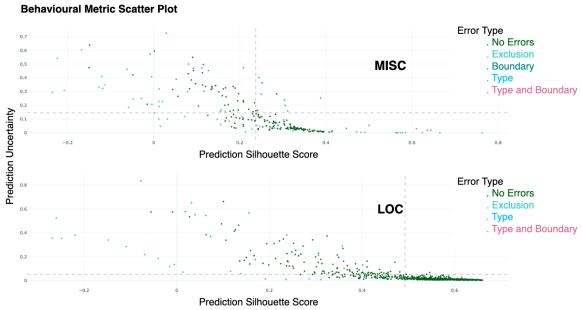

Figure 6: Relationship between predicted silhouette score (x-axis) and prediction uncertainty (y-axis).

Anomaly detection: Interactive linking reveals two distinct error patterns highlighted in Figure 5. *Region C* contains correctly predicted `O` tokens with interestingly low predicted and true silhouette scores. Although predicted as non-entities, their embeddings lie close to entity clusters, indicating representation–output misalignment. Instance-level inspection confirms these are annotation inconsistencies—tokens labelled `O` despite appearing in entity-like contexts. The model encodes semantic structure in the representation space while the output layer reproduces noisy training labels. Additional examples and instance-level evidence are provided in Appendix A.12.

Region A forms a dense cluster of mixed true negatives and systematic errors. Investigation using the Token Context View reveals that many tokens appear abruptly at sentence boundaries without semantic relevance. Comparison between the original Benajiba corpus and the CAMeL Lab version shows these tokens were displaced during dataset standardisation, likely from preprocessing or sentence segmentation errors. Supporting evidence and examples are provided in Appendix A.13.

These two anomaly types exhibit distinct behavioural signatures (Figure 7). Systematic preprocessing errors (Region A, diamond markers selected from Figure 5) are characterised by high confidence, high loss, very low or negative silhouette scores, and low uncertainty—the model is confidently wrong. In contrast, annotation-related errors (Region C) show high uncertainty and moderate

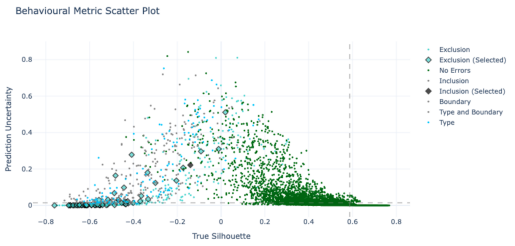

Figure 7: Error types show distinct signatures. Annotation errors (high uncertainty) vs. systematic errors (low uncertainty, poor separation).

confidence—the model is unsure. Together, these patterns explain the calibration failures observed in Stage 1.

4 Related Work

DeformAR sits at the intersection of interpretability, visual analytics for NLP models, and NER evaluation. Interpretability methods are commonly categorised as global or local (Zini and Awad, 2023; Ferrando et al., 2024). Global approaches analyse corpus- or representation-level behaviour, often via dimensionality reduction or probing (Aken et al., 2019), while local approaches explain individual predictions using attention or attribution techniques. However, attention-based explanations remain contested (Sun et al., 2021), and attribution methods such as LIME or SHAP are difficult to adapt reliably to sequence labelling tasks (Ruder et al., 2022). DeformAR bridges evaluation and interpretability by embedding diagnostic capabilities directly into evaluation. While traditional interpretability methods explain what the model learned and evaluation metrics measure aggregate performance, DeformAR investigates why specific errors occur through coordinated analysis of data properties, learned representations, and prediction outcomes. This diagnostic approach reveals not just that the model fails, but how data characteristics, tokenization artifacts, and representation quality interact to produce specific error patterns.

Several prior works analyse model behaviour through behavioural signals such as loss, confidence, or prediction consistency. Dataset Cartography (Swayamdipta et al., 2020) characterises training dynamics to identify easy and hard examples, while slice- and bucket-based evaluations partition data by interpretable attributes such as frequency or span length (Fu et al., 2020; Liu et al., 2021). These approaches provide valuable signals but typically focus on either training dynamics or dataset properties in isolation. In contrast, DeformAR integrates behavioural metrics across data, representation, and output components, and links them through interactive analysis at inference time.

Visual analytics systems support inspection of Transformer models, including LIT (Tenney et al., 2020), InterpreT (Lal et al., 2021), and T3-Vis (Li et al., 2021). These tools differ in temporal focus: T3-Vis tracks training dynamics (attention head evolution, learning trajectories), while LIT and InterpreT analyse inference-time behaviour through counterfactuals and layer-wise attention inspection. DeformAR differs from all three in being evaluation-driven and task-specific. Unlike T3-Vis's focus on how models learn, DeformAR diagnoses why trained models fail by linking data properties, learned representations, and prediction outcomes. Unlike LIT/InterpreT's general-purpose exploration, DeformAR integrates NER-specific metrics (span-level errors, BIO structure) with behavioural signals to support targeted diagnostic workflows. A detailed **comparison of DeformAR with existing visual analytics tools is provided** in Appendix A.14 (Tables 5, 3, and 4).

Recent work on NER evaluation has highlighted the impact of annotation errors and dataset artifacts. CLEANANERCorp (AlDuwais et al., 2024) and similar efforts for CoNLL-2003 (Liu and Ritter, 2023; Rücker and Akbik, 2023) focus on correcting annotation inconsistencies through semi-automated methods. DeformAR complements these approaches: rather than performing correction, it uncovers similar issues through interactive analysis and provides explanatory insight into how such errors propagate through representations and model behaviour, supporting targeted remediation.

5 Discussion and Future Work

This work argues for a shift in how Arabic NER systems are evaluated and interpreted. Rather than treating evaluation as a final step that reports aggregate metrics, DeformAR embeds interpretability directly into the evaluation process through component-level diagnosis. Our analysis shows that many performance failures—such as recall degradation, calibration errors, or span-level boundary mistakes—cannot be understood from precision and recall alone, but emerge from interactions between data properties, learned representations, and output-layer behaviour.

Our case study revealed three key findings. First, we identified two distinct error types that look the same in aggregate metrics but have different root causes: annotation inconsistencies (high uncertainty, moderate confidence) and systematic preprocessing errors (high confidence, high loss, negative silhouette). These require different fixes—relabelling versus data cleaning.

Second, **interactive filtering enabled isolation of subcomponent impact such as tokenization effects from inherent data properties.** By comparing token-level metrics before and after WordPiece tokenization, we found that clearly annotated tokens became ambiguous or inconsistent due to distinct words sharing first subwords. This finding—that tokenization introduces new ambiguity rather than exposing existing data issues—has implications for morphologically rich languages using subword tokenization and demonstrates the value of coordinated component analysis.

Third, **representation quality and model confidence do not always align with correctness.** Multi-resolution analysis showed that MISC underperforms due to compounding factors: low frequency, high lexical diversity, longer spans, and noisy annotations. These interact to produce weak representations and calibration failures. Aggregate metrics cannot trace this chain from data properties through representations to prediction errors.

DeformAR is **intentionally diagnostic rather than corrective**. While it does not propose automatic mitigation methods, it provides the necessary evidence to support informed intervention. In low-resource settings, where annotation revision and model retraining are costly, understanding *why* a system fails is often a prerequisite for deciding *how* to improve it. The framework offers both global interpretability (through corpus-level patterns and representation structure) and local interpretability (through token- and instance-level inspection), grounded in behavioural evidence rather than attribution scores.

Our focus on a single dataset and model configuration was a deliberate choice: we prioritised diagnostic depth over comparative breadth. Multi-model comparisons risk attributing differences to architecture without understanding root causes. By analyzing one configuration in detail, we isolated how data, model, evaluation subcomponents interact to produce errors—insights that shallow multi-model benchmarking would obscure.

Future work could extend the framework in several directions. First, integrating semi-automated correction workflows—such as targeted relabelling or data augmentation guided by the diagnostic insights. Second, comparative analysis across model architectures, output layers, and cross-lingual transfer settings could reveal architectural sensitivities invisible in single-model evaluation. Third, extending beyond inference-time analysis to track training dynamics would enable diagnosis of when and why errors emerge during learning. Finally, while our case study focuses on Arabic NER, the challenges here are amplified but not unique to Arabic. We plan to extend the analysis to other languages, datasets (including cleaned versions of ANERCorp), and sequence labelling tasks beyond NER. DeformAR provides a general diagnostic template for structured evaluation of sequence labelling systems, particularly in low-resource and morphologically rich languages.

6 Conclusion

We presented DeformAR, a diagnostic visual analytics framework for evaluating Arabic NER systems through cross-component analysis. By integrating token-level behavioural metrics with interactive visualizations, DeformAR enables systematic exploration of errors, representation structure, and model behaviour beyond aggregate metrics. Through a case study on ANERCorp, we demonstrated how DeformAR uncovers distinct failure modes—annotation inconsistencies versus preprocessing artifacts—and revealed how tokenization introduces ambiguity beyond existing data issues. Multi-resolution analysis traced MISC's underperformance to compounding factors that interact to produce weak representations and calibration failures. By embedding interpretability directly into evaluation, DeformAR bridges performance measurement and explanation, providing a foundation for diagnostic evaluation in Arabic and other low-resource settings.

Limitations

DeformAR is designed as a diagnostic framework rather than a corrective or performance-optimising method. While it identifies root causes of model errors—such as annotation inconsistencies, tokenisation-induced ambiguity, and representation–output misalignment—it does not automatically apply fixes. Addressing these issues (e.g., relabelling data, revising preprocessing, or modify-

ing model architectures) requires human judgment and domain expertise.

In this sense, DeformAR differs from data-cleaning approaches such as CleanANERCorp, which primarily rely on model confidence signals to identify potentially noisy annotations and guide subsequent manual correction. Rather than centring intervention on a single signal or correction mechanism, DeformAR aims to provide a holistic diagnostic view of interacting system components, supporting more informed and targeted decisions about where and how intervention may be most effective.

Our analysis focuses on inference-time behaviour and does not currently incorporate training dynamics. Although this allows precise attribution of errors to interactions between data properties, learned representations, and output behaviour, it does not capture how or when such issues emerge during learning. Extending DeformAR to integrate training-time signals is an important direction for future work.

The case study examines a single dataset and model configuration. This choice was deliberate: we prioritised diagnostic depth over comparative breadth in order to isolate and explain specific failure mechanisms without confounding architectural differences. As a result, the empirical findings should not be interpreted as universal properties of Arabic NER models. However, the diagnostic methodology—component-level analysis linked through behavioural and representation-level evidence—is model- and language-agnostic.

Finally, effective use of DeformAR requires language-specific expertise for instance-level interpretation, particularly in morphologically rich languages such as Arabic. While the framework surfaces anomalies and patterns automatically, understanding their linguistic or annotation-related causes depends on human inspection. This human-in-the-loop requirement reflects an intentional design trade-off common to visual analytics systems.

Acknowledgments

I would like to thank my supervisors, Julie Weeds and David Weir, for their guidance and support throughout this work.

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.

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.

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 Core Token Illustration and Subcomponent Interactions

Figure 8 illustrates how tokenisation creates the core token mechanism and how subcomponents interact during fine-tuning. We use two examples from ANERCorp: the word *Al-Tarawina* (a location name) and *Al-Salihia* (another location).

Core tokens and gradient updates. When a word is tokenised into multiple subwords (e.g., *Al-Tarawina* → *Al-Tar*, *awina*), only the first subword becomes the **core token** and receives the entity label (here, B-LOC). The remaining subwords are marked as IGNORED. "Ignored" specifically

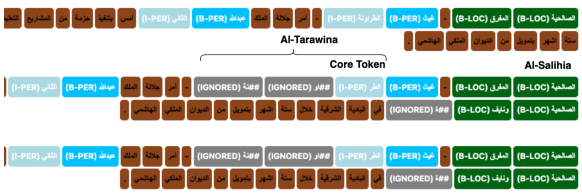

Figure 8: Example of core token assignment and subcomponent interactions. The word *Al-Tarawina* is tokenised into multiple subwords, but only the first subword (*Al-Tar*) is designated as the core token and assigned the entity label B-LOC. The remaining subwords are marked as IGNORED for loss computation and gradient updates, though they contribute to contextual representations.

means these tokens are excluded from loss computation and gradient updates—they do not influence parameter optimization during backpropagation. However, they are *not* ignored during forward computation: all subwords contribute to generating contextualized representations in the encoder.

Subcomponent interactions during fine-tuning. Several interactions occur between model and data subcomponents:

1. Output layer ↔ True labels: The output layer learns to map representations to entity labels based on supervision from the training data. However, this supervision is filtered through the core token mechanism—only core token positions receive gradient signals.

2. Loss function ↔ Predictions and labels: The loss function compares predicted labels against true labels, but only for core tokens. If the predicted distribution diverges from the true label, the loss increases, and backpropagation adjusts model parameters accordingly. Non-core tokens contribute to the forward pass (generating representations) but are excluded from this optimization loop.

3. Representation layer ↔ All subwords: The representation layer (BERT encoder) processes *all* subwords to generate contextualized embeddings. For example, the core token *Al-Tar* receives a representation that is contextualized by the surrounding subwords (*awina*) and other tokens in the sentence. However, when these representations are fed to the output layer, only the core token's representation is used to predict the entity label and compute loss.

4. Vocabulary (core tokens) ↔ True labels: The relationship between core tokens and their assigned labels exhibits several measurable properties. *Overlap* occurs when the same core token appears with different entity tags across the dataset (e.g., a token labelled as both B-LOC and O in different contexts). *Ambiguity* measures how uncertain a token's label assignment is based on its training distribution. *Inconsistency* captures disagreement between training and test labels for the same token.

Why this matters for evaluation. These interactions occur internally during training and inference but are not directly observable through standard evaluation metrics like F1 score. DeformAR extracts these subcomponents and characterizes their behaviour using token-level metrics (Section 2.1), enabling systematic diagnosis of how tokenization, representation quality, and label assignments jointly affect performance.

A.2 Metric Definitions

This appendix provides brief definitions of the behavioural metrics used throughout the paper.

Lexical metrics Lexical diversity is measured using the type-to-token ratio. Out-of-vocabulary (OOV) rate measures the proportion of test tokens that never appeared with the same entity tag during training.

Annotation metrics **Ambiguity** measures how often a token appears with multiple entity tags in the training data. Formally, a token is considered ambiguous if it is associated with more than one entity tag (including `O`) across its training occurrences. **Label inconsistency** measures disagreement between training and test labels for the same token. **Entity tag overlap** quantifies the proportion of token types shared between different entity tags (e.g., `O` and `ORG`).

Model metrics Prediction confidence is defined as the maximum softmax probability. Uncertainty is measured using Shannon entropy over the predicted label distribution. Token-level loss is computed using cross-entropy. **Representation separability** is assessed using silhouette scores computed over contextual token embeddings, using either the true or predicted entity labels as cluster assignments.

Span-level error categorisation Span-level errors are categorised by extending the `seqeval` evaluation pipeline. We distinguish: (i) **exclusion errors**, where a gold entity span is predicted entirely as `O`; (ii) **inclusion errors**, where a predicted entity

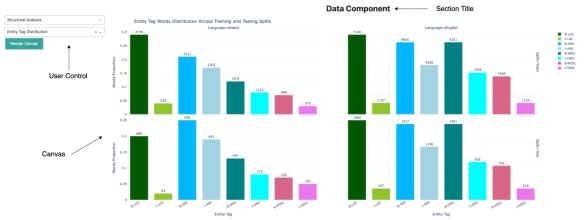

Figure 9: Layout of the Cross-Component View in DeformAR. The interface consists of three main elements: (1) a *user control panel* (top-left) for selecting analysis types and visualisations; (2) *component canvases*, where each canvas corresponds to a system component (e.g., Data, Model, Evaluation) and displays the selected metrics; and (3) *section headers* that label each component and align plots across dataset splits or languages. This design supports side-by-side, metric-consistent comparison across components.

span does not correspond to any gold span; (iii) **boundary errors**, where a predicted span overlaps with a gold span of the same entity type but has incorrect boundaries; and (iv) **type errors**, where a predicted span overlaps with a gold span but is assigned an incorrect entity type. This categorisation enables analysis beyond flat token-level confusion matrices.

A.3 Cross-Component View Interface and Configuration

Figure 9 illustrates the layout of the Cross-Component View. The interface is organised into three conceptual elements: *user controls*, *component canvases*, and *section headers*.

User controls The control panel (top-left) allows users to select the analysis type (e.g., structural statistics, behavioural metrics) and the specific visualisation to be rendered. These selections determine which metrics are displayed across all canvases, enabling consistent comparison across components.

Component canvases Each NER component (Data, Model, Evaluation) is rendered on a dedicated canvas. Canvases display the selected metrics using standard visual encodings such as bar charts, heatmaps, or tables. This design supports direct comparison across components and across dataset splits or languages within the same view.

Section headers and alignment Each canvas is labelled with a section header indicating the component being analysed (e.g., *Data Component*).

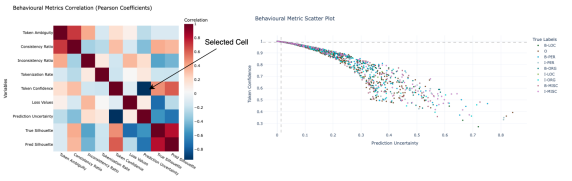

Figure 10: Metric Correlation Heatmap dynamically linked to the Behavioural Scatter Plot. Selecting a heatmap cell updates the scatter plot axes to the corresponding metric pair.

Within a canvas, plots are vertically aligned to reflect dataset splits (e.g., training vs. test), ensuring structural consistency across components.

Configuration and extensibility The available analyses, plot types, and component layouts are defined declaratively via a YAML configuration file. Adding a new visualisation requires implementing the corresponding plotting logic and registering it in the configuration, after which it becomes selectable through the user controls. This design allows the interface to be extended without modifying the core dashboard logic.

A.4 Behavioural Analysis View: Interaction Design

This appendix provides implementation-level details of the Behavioural Analysis View, focusing on how interactive linking supports exploratory analysis across behavioural metrics and representation structure.

Metric Correlation Heatmap and Dynamic Axis Linking. Figure 10 shows the Metric Correlation Heatmap alongside the Behavioural Scatter Plot. Each cell in the heatmap represents the Pearson correlation between a pair of behavioural metrics (e.g., ambiguity, confidence, loss, uncertainty). Selecting a cell dynamically assigns the corresponding metric pair to the x- and y-axes of the Behavioural Scatter Plot. This enables rapid, targeted inspection of specific metric interactions without manual reconfiguration.

Brushing and Bidirectional Linking. As shown in Figure 11, users can apply rectangular or lasso-based brushing in either the Behavioural Scatter Plot or the UMAP Projection. Selections are propagated bidirectionally: tokens selected in one view are highlighted in all linked views. This enables users to trace subsets of tokens across be-

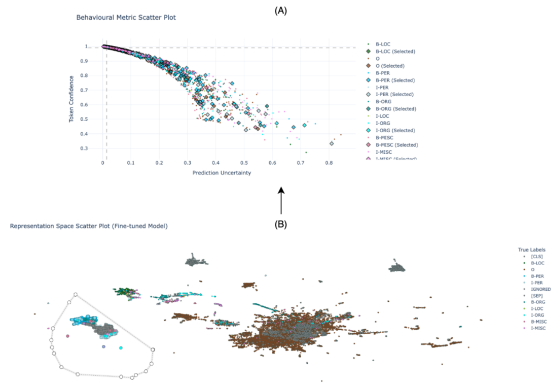

Figure 11: Bidirectional brushing between the Behavioural Scatter Plot and the UMAP Projection. Selected tokens are synchronised across views.

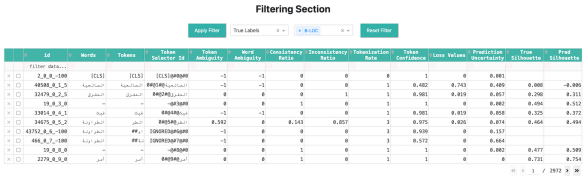

Figure 12: Global filtering panel for constraining analysis by categorical and numerical attributes. Filters are applied consistently across all linked views.

havioural space and representation space, supporting anomaly discovery and hypothesis validation.

Filtering Panel. Figure 12 shows the filtering interface, which allows users to constrain the active dataset by categorical (e.g., label, error type) or numerical (e.g., confidence thresholds) attributes. Filters apply globally across all linked views, enabling focused analysis of specific subsets such as a single entity type or error category.

Selection Summary and Aggregation. Figure 13 presents the Selection Summary panel, which aggregates statistics for the currently selected subset. This includes categorical summaries (e.g., error type distribution, label agreement) and numerical metric summaries (e.g., mean, quartiles, extrema). The summary supports rapid validation of hypotheses formed through visual inspection.

A.5 Instance-Level View

The Instance-Level View supports example-based inspection at the sentence and token levels, enabling users to ground previously identified behavioural patterns in concrete model predictions. This view is used extensively throughout the analysis to validate and interpret observations from

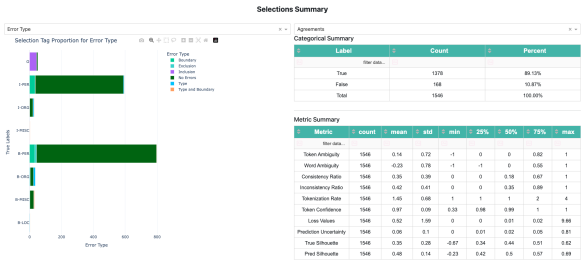

Figure 13: Selection Summary panel showing categorical distributions and aggregated behavioural metrics for the active selection.

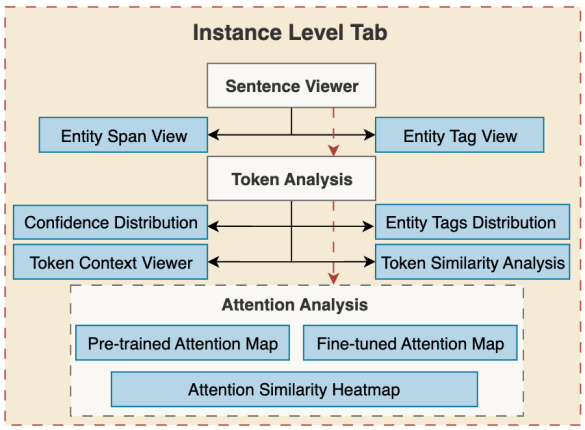

Figure 14: Instance-Level View supporting sentence-level inspection, token-level analysis, and attention visualisation.

higher-level views; to avoid redundancy, we do not provide exhaustive qualitative examples here.

Figure 14 shows the overall layout of the Instance-Level View, which integrates sentence-level annotation inspection, token-level behavioural summaries, and attention-based visualisation.

As an illustrative example, Figure 15 presents an attention similarity heatmap comparing pretrained and fine-tuned models. The heatmap shows how attention patterns for individual heads and layers change after task-specific fine-tuning, highlighting which components are most affected by adaptation. This example demonstrates how the Instance-Level View can be used to relate behavioural or representation-level findings to internal model mechanisms without relying on attention as a standalone explanation.

A.6 Training and Evaluation Details

This appendix documents the training configuration and evaluation protocol used in the case study presented in Section 3.

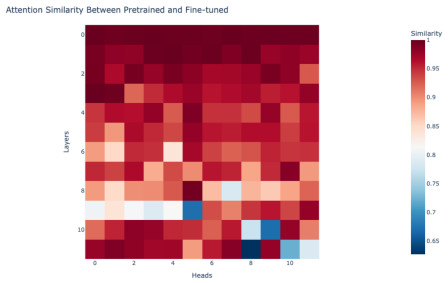

Figure 15: Attention similarity between pretrained and fine-tuned models, computed per layer and head. Lower similarity indicates stronger task-specific adaptation.

Model and training setup We fine-tune AraBERTv02-base, a 12-layer Transformer model with 768 hidden units, using a linear token classification head. Training is performed using the AdamW optimiser with a learning rate of `5e-5` and a batch size of 16 for 4 epochs. A linear learning rate scheduler with warm-up is applied, using a warm-up ratio of `0.1` to stabilise early training.

Dropout with a rate of `0.1` is applied before the classification layer. Gradient accumulation is set to 1, and gradient clipping with a maximum norm of `1.0` is used to prevent unstable updates. All model parameters are fine-tuned except LayerNorm and bias terms, which are frozen following common practice to improve stability and reduce over-fitting.

Tokenisation and labelling Input text is tokenised using the WordPiece tokenizer associated with AraBERTv02. Original word-level IOB2 annotations are aligned to subword tokens by assigning the original tag to the first subword and propagating the corresponding inside tag to subsequent subwords. Special tokens (e.g., `[CLS]`, `[SEP]`) are excluded from loss computation and evaluation.

Evaluation protocol All NER evaluations are conducted using the `seqeval` library (Nakayama, 2018) in *strict* mode with the IOB2 tagging scheme. In strict mode, a predicted entity is counted as correct only if both its span boundaries and entity type exactly match the gold annotation. This ensures that boundary errors, type errors, and exclusion errors are penalised appropriately and prevents partial span matches from inflating performance scores.

The same evaluation configuration is used consistently across all reported experiments. Span-level error categories used in DeformAR (boundary, exclusion, inclusion, and type errors) are derived by extending the standard `seqeval` output, enabling structured analysis beyond flat token-level confusion matrices.

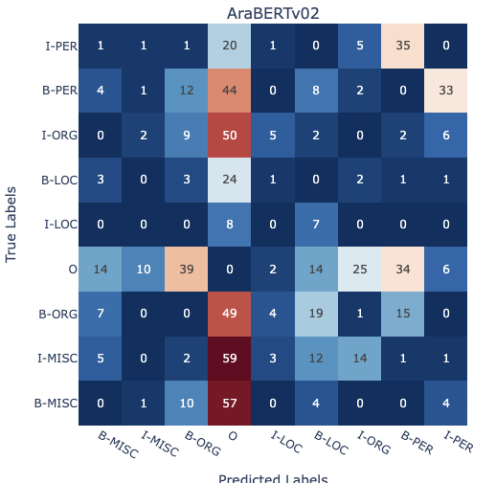

Figure 16: Flat token-level confusion matrix for AraBERTv02 on ANERCorp using IOB2 tags.

A.7 Token-Level Confusion Analysis

Figure 16 presents the flat token-level confusion matrices for AraBERTv02 using IOB2 tags (e.g., B-LOC, I-LOC, B-PER). This view complements the span-level error analysis by providing fine-grained evidence of how different error types manifest at the token level.

Exclusion errors manifest as entity tokens (particularly B-MISC and I-MISC) being predicted as `O`. Boundary errors appear as confusions between beginning and inside tags of the same entity type (e.g., B-PER ↔ I-PER), while type confusions are visible between semantically related categories such as ORG and LOC. These token-level patterns underlie the aggregated span-level error categories presented in Figure 4.

A.8 Data Characteristics

This appendix provides supporting evidence for the data-side analysis discussed in Section 3, focusing on entity frequency, lexical structure, annotation quality, and span complexity. Figure 17 shows the distribution of entity tags across training and test splits, confirming strong class imbalance, with LOC being the most frequent and MISC the least. However, frequency alone does not explain performance differences.

Lexical sparsity and coverage are illustrated in Figures 18 and 19. MISC exhibits the highest out-of-vocabulary rates and substantial overlap with

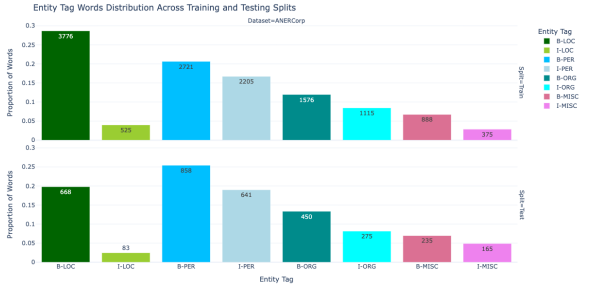

Figure 17: Distribution of entity tag across training and test splits.

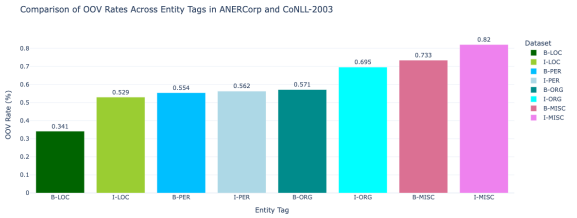

Figure 18: OOV rates by entity tag, showing the proportion of token types in the test set not seen with the same tag in training.

the O tag, indicating weak lexical grounding and increased confusion with non-entity tokens. In contrast, LOC benefits from repeated exposure to a smaller set of token types despite similar overlap with O.

Figures 20, 21, and 22 characterise lexical distributions in more detail. LOC shows a highly skewed token-type frequency distribution dominated by a few frequent types, whereas MISC exhibits a flatter, long-tailed distribution with many rare types and higher lexical diversity, both before and after tokenisation. Tokenisation reduces diversity only marginally and does not mitigate sparsity for MISC.

Structural properties of entities are shown in Figure 23. LOC spans are shorter and simpler, while MISC spans are longer and more complex, increasing sensitivity to boundary and exclusion errors. Finally, annotation quality is examined in Figure 24, which shows that ambiguity and inconsistency are concentrated in I-tags for LOC but affect both B- and I-tags for MISC. This asymmetry reduces the impact of inconsistency for LOC while amplifying error propagation for MISC.

Taken together, these figures show that LOC benefits from high frequency, lexical repetition, shorter spans, and cleaner annotations, while MISC suffers from sparsity, high ambiguity, longer spans, and inconsistent labelling. These data properties provide a foundation for the behavioural and representation-level patterns observed in later analysis.

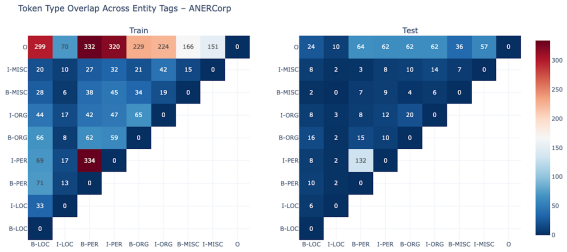

Figure 19: Entity tag overlap matrix showing the number of token types associated with multiple tags in training and test sets.

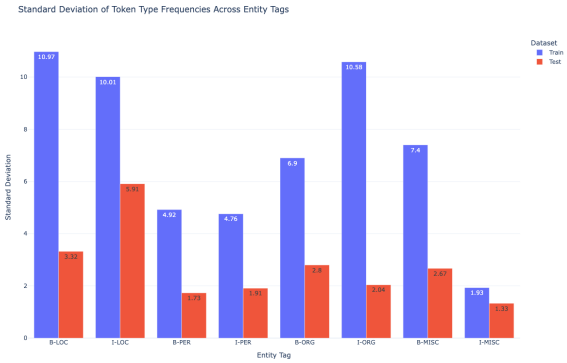

Figure 20: Standard deviation of token type frequencies across entity tags in training and test splits. For each entity tag, we compute how often each token type appears and calculate the standard deviation across those frequency counts. Higher values indicate skewed distributions with a few highly frequent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

A.9 Model Behaviour Evidence

This appendix provides visual evidence supporting the model behaviour analysis discussed in Section 3, focusing on loss, calibration, confidence, representation structure, and error manifestation across entity tags.

MISC exhibits systematically higher token-level loss than other entity types. As shown in Figure 25, I-MISC has the highest mean loss and variance, indicating persistent difficulty during inference even after fine-tuning. In contrast, B/I-LOC maintain consistently low loss values, reflecting stable and predictable model behaviour.

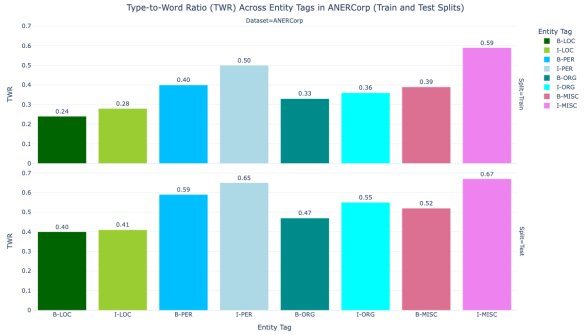

Figure 21: Lexical diversity (type-to-word ratio) across entity tags before tokenisation.

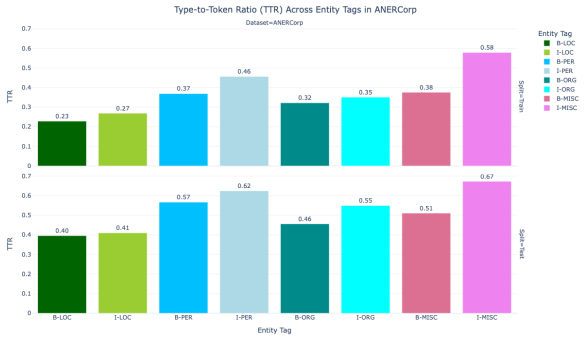

Figure 22: Lexical diversity (type-to-token ratio) across entity tags after tokenisation. The change due to tokenisation is minimal for most tags, with PER being the most affected.

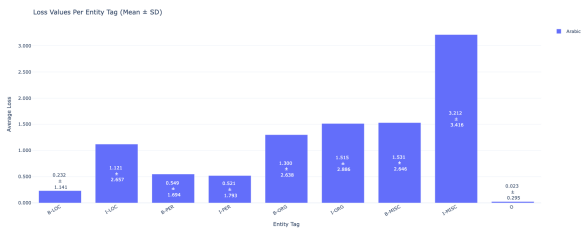

Figure 25: Mean token-level loss per entity tag (mean ± SD).

Calibration differences are further exposed through prediction uncertainty. Figure 26 shows that for LOC, uncertainty is low when predictions are correct and substantially higher when they are incorrect, indicating well-calibrated behaviour. In contrast, MISC exhibits elevated uncertainty even for correct predictions, with limited separation between correct and incorrect cases—particularly for B-MISC and I-MISC—indicating calibration failure rather than isolated misclassification.

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

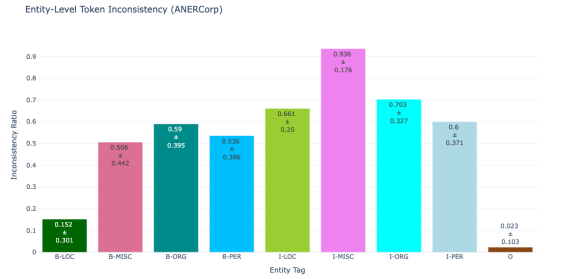

Figure 24: Token-level inconsistency ratio across entity tags. High values indicate that tokens are associated with multiple labels in the training data.

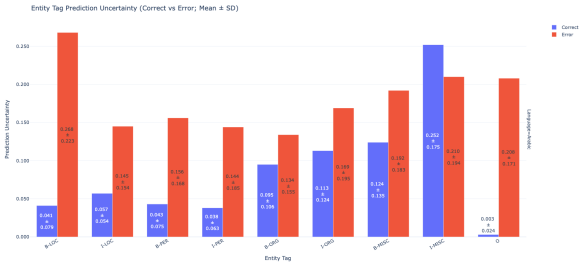

Figure 26: Prediction uncertainty for correct vs. incorrect predictions by entity tag (mean ± SD).

Confidence-based analysis reveals that many MISC exclusion errors (MISC→O) occur with high confidence. As shown in Figure 27, a large proportion of confidence mass for B/I-MISC is assigned to the O label, indicating confident exclusion rather than uncertainty-driven prediction. LOC errors, by contrast, are associated with lower confidence and more conservative predictions.

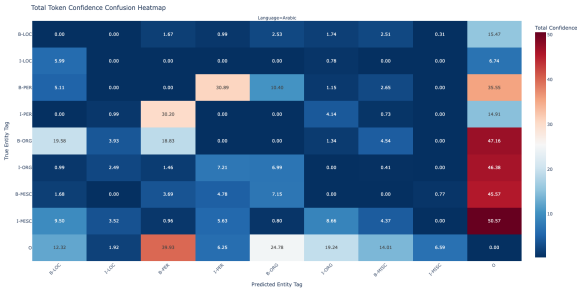

Figure 27: Confidence-weighted confusion matrix showing total prediction confidence mass across true and predicted entity tags.

Representation-level evidence reinforces these behavioural patterns. Figure 28 shows that B/I-LOC tokens achieve consistently high silhouette scores under both true and predicted labels, indicating well-separated embeddings. In contrast, I-MISC exhibits near-zero or negative silhouette scores, and B-MISC shows weak separation, reflecting substantial overlap with O and other entity types in representation space.

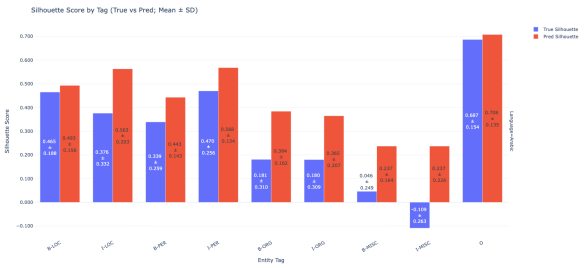

Figure 28: Silhouette scores by entity tag computed over contextual embeddings (true vs. predicted labels).

Overall, Figures 25–28 collectively show that MISC underperformance arises from weak representation structure, poor calibration, and high-confidence exclusion errors, whereas LOC benefits from well-separated embeddings, reliable confidence behaviour, and lower exclusion rates.

A.10 Tokenisation Impact

This appendix provides supporting evidence for the tokenisation effects discussed in Section 3. We analyse how subword tokenisation alters lexical structure, annotation consistency, and error patterns in Arabic NER.

At the corpus level, tokenisation increases overlap between entity tags. Figures 29 and 30 compare word-level and token-level type overlaps across entity tags in the training and test splits. While word-level overlap is already substantial—especially between entity tags and O—tokenisation amplifies this effect by collapsing distinct words into shared subword units. This increased overlap is most pronounced for MISC and ORG and aligns with the high rate of exclusion errors observed for these spans.

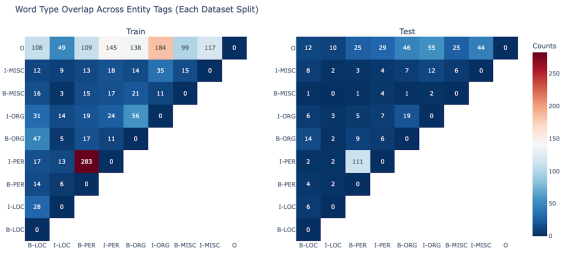

Figure 29: Word-level type overlap across entity tags in the training and test splits.

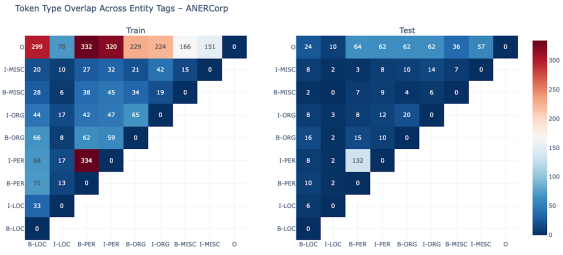

Figure 30: Token-level type overlap after WordPiece tokenisation.

Despite the increase in tag overlap, tokenisation has a limited effect on overall lexical diversity. Figures 31 and 32 show that type-to-word and type-to-token ratios remain largely stable across entity tags. The most noticeable reduction occurs for PER, reflecting the frequent decomposition of personal names into common subword fragments.

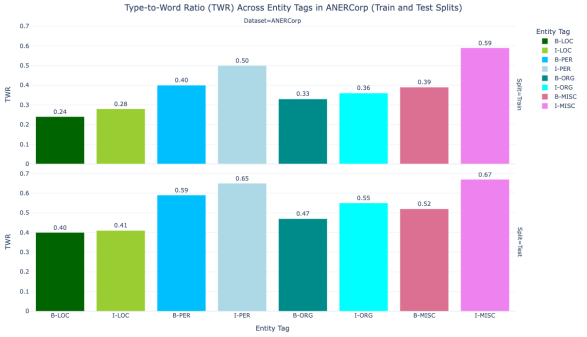

Figure 31: Word-level lexical diversity (type-to-word ratio) across entity tags.

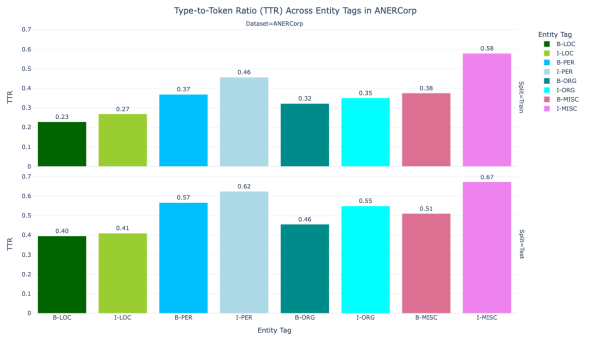

Figure 32: Token-level lexical diversity (type-to-token ratio) after tokenisation.

Beyond aggregate statistics, instance-level inspection reveals how tokenisation and orthographic ambiguity introduce annotation noise. Figure 33 shows a case of diacritic ambiguity, where two surface-identical words correspond to different meanings ("Spanish" vs. "Spain") but receive conflicting supervision due to the absence of diacritics. One instance is incorrectly labelled as `B-LOC`, while the other is correctly annotated.

Exposure to such patterns causes a systematic mismatch between model components: the output layer learns to predict `B-LOC` based on recurring annotation patterns, whereas the representation layer encodes the token according to its contextual semantic meaning. This divergence exposes a representation–output misalignment, where correct semantic encoding coexists with label-driven misprediction.

Figure 33: Inconsistent labelling caused by diacritic ambiguity in Arabic.

Figure 34 illustrates subword-induced ambiguity arising from WordPiece tokenisation. The first subword of a longer place name overlaps with a country name observed during training, causing conflicting supervision at the subword level. In this example, the token *Kat*—extracted from the word *Catalonia*—is misclassified as `B-LOC` due to its overlap with training instances where the same subword appears in the country name *Katanga*.

While the contextual representation correctly places the token in a non-entity (`O`) region of embedding space based on sentence context, the output layer predicts `B-LOC` by reproducing learned annotation patterns. This again exposes a representation–output misalignment, demonstrating how subword tokenisation can decouple semantic encoding from prediction behaviour.

Figure 34: Ambiguity introduced by shared subwords across entity labels.

A.11 Behavioural Correlation Analysis

This appendix provides supporting evidence for the span-specific behavioural analysis presented in Section 3, focusing on differences between high-performing (LOC) and low-performing (MISC) spans. We report correlations and scatter plots relating prediction confidence, uncertainty, and representation separability (silhouette scores).

LOC span behaviour Figure 35 shows the Pearson correlation matrix for behavioural metrics restricted to LOC tokens. A strong positive correlation is observed between prediction confidence and true silhouette score, consistent with the main analysis: confident LOC predictions tend to correspond to well-separated representations.

Figure 36 further illustrates this relationship through a scatter plot of confidence versus true silhouette score for LOC tokens. Correct predictions cluster above the mean confidence and silhouette thresholds (indicated by dotted reference lines), while errors are concentrated in the low-confidence, low-separation region. This reflects stable calibration and alignment between representation structure and model outputs for LOC.

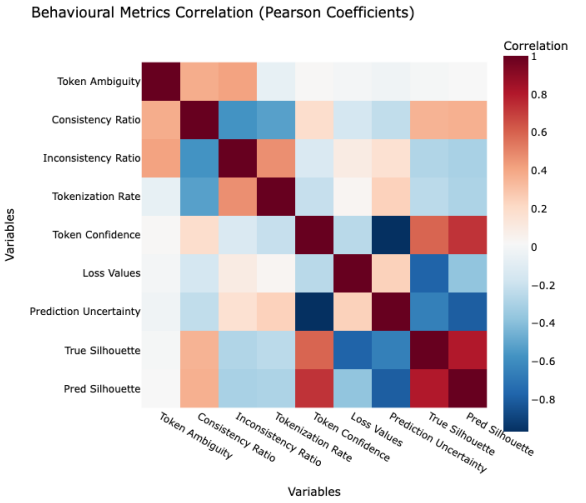

Figure 35: Correlation heatmap of behavioural metrics for LOC tokens. Strong positive correlation is observed between confidence and true silhouette score.

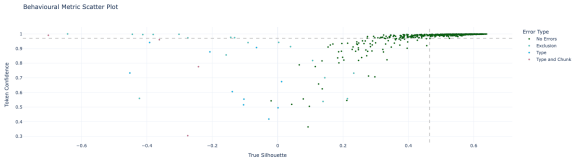

Figure 36: Scatter plot of confidence versus true silhouette score for LOC tokens. Dotted lines indicate mean values. Correct predictions cluster in the high-confidence, high-separation region.

MISC span behaviour In contrast, Figure 37 shows that correlations between behavioural metrics are substantially weaker for MISC tokens. In particular, confidence exhibits only a weak relationship with true silhouette score, indicating misalignment between prediction certainty and representation quality.

This misalignment is illustrated in Figure 38, which plots confidence against true silhouette score for MISC. Exclusion errors (entity → 0) frequently appear above the mean confidence threshold while exhibiting low or negative silhouette scores. Conversely, some correct MISC predictions fall below average confidence despite moderate separability. Additionally, true silhouette scores for MISC are overall lower than for LOC, reflecting weaker representation structure.

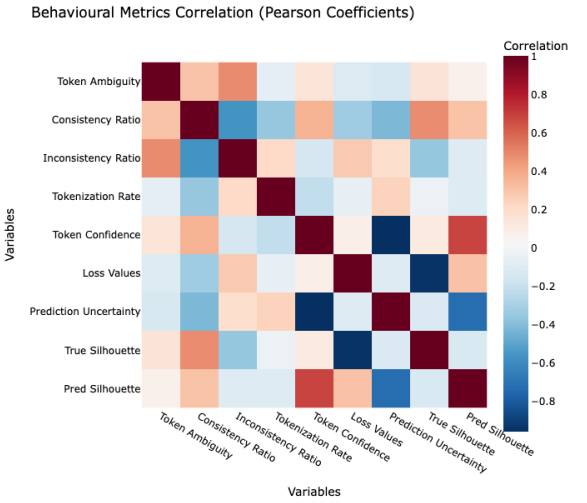

Figure 37: Correlation heatmap of behavioural metrics for MISC tokens. Relationships between confidence, uncertainty, and silhouette scores are markedly weaker than for LOC.

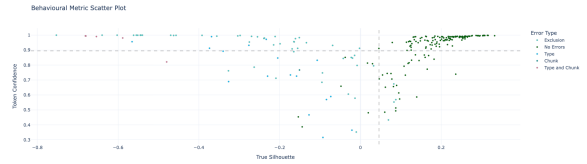

Figure 38: Scatter plot of confidence versus true silhouette score for MISC tokens. High-confidence exclusion errors and low-confidence correct predictions highlight calibration failures.

A.12 Annotation Errors: Region C Analysis

This appendix demonstrates how DeformAR's interactive linking enables the discovery and diagnosis of annotation errors through a detailed walkthrough of Region C, identified in Section 3. We trace the investigation from initial pattern detection through behavioural metrics to instance-level evidence.

Step 1: Identifying the anomaly through behavioural scatter Figure 39 (Top) shows the behavioural scatter plot of predicted silhouette versus true silhouette scores. Region C (highlighted by the dashed box) (Middle) contains tokens with unusually low scores for both metrics, indicating poor representation separability under both true and predicted labels. Interestingly, these tokens are *correctly predicted* as O, raising the question: why do correctly predicted tokens exhibit such poor representation quality?

Step 2: Visualizing spatial distribution in UMAP Using bidirectional linking, the tokens selected in Region C are highlighted in the UMAP projection (Figure 39). Rather than clustering in the dominant O region, these tokens (brown points) are embedded among B-PER clusters (blue points). This spatial placement reveals a key insight: the representation layer has encoded these tokens as semantically similar to person entities, despite their O predicted labels.

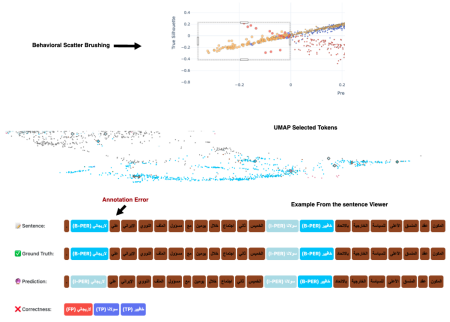

Figure 39: UMAP projection showing Region C tokens (Brown Diamonds) embedded within the B-PER cluster rather than the O region, indicating that representations reflect semantic content rather than training labels.

Step 3: Instance-level inspection reveals annotation errors To understand why these tokens exhibit this behaviour, we examine specific instances using the Sentence Viewer. Figure 39 (Bottom) shows an example sentence containing the token (*Ali*), a common Arabic name. The sentence annotation (Ground Truth) labels this token as O (non-entity), while it should have been labelled as B-PER. The model's prediction (third row) predicts the token as O and the token next to it as I-PER.

To understand why the output layer predicts O despite the representation suggesting B-PER, we examine the training distribution of the token (Figure 40). The token appears predominantly as O in the training data (over 200 occurrences) compared to fewer than 50 occurrences as B-PER and negligible occurrences as I-PER or I-MISC. The output layer has memorized this statistical pattern from training rather than learning to distinguish semantic entity types. When the token appears in the test set, the model confidently predicts O—the dominant training label—even though the contextual representation encodes person-like semantics.

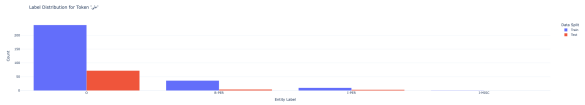

Figure 40: Label distribution for the token "Ali" across training (blue) and test (red) splits. The token appears predominantly as O in training, explaining why the output layer predicts this label despite semantic evidence for B-PER.

This creates the observed representation-output misalignment:

- The representation layer encodes as semantically similar to person names (hence its placement in the B-PER cluster in UMAP).
- The output layer predicts O because it learned from the training label distribution for this token.
- The result is low silhouette scores: the token is far from its predicted label cluster (O) and embedded in its true semantic cluster (B-PER).

Implications This analysis reveals a fundamental issue in the training data: annotation inconsistency where the same token receives different labels depending on context, with one label (here, O) dominating. The model's two subcomponents respond differently:

- The **representation layer** learns semantic patterns from the full context, placing "Ali" near other person names.
- The **output layer** learns label distributions from supervision, predicting the statistically dominant label O.

This representation-output misalignment is invisible to aggregate metrics (the prediction is technically "correct" according to the noisy labels) but is surfaced by DeformAR through the combination of behavioural metrics (low silhouette scores), spatial visualization (UMAP placement), and instance-level inspection (sentence context). Such cases highlight the value of diagnostic evaluation beyond F1 scores: they indicate data quality issues that, if corrected, could improve both model calibration and true performance.

A.13 Examples of Systematic Preprocessing Errors

This section provides qualitative evidence for systematic preprocessing errors identified during be-

haviour analysis, specifically sentence-start misalignment artifacts corresponding to Region A in Figure 5.

Sentence-start misalignment Figure 41 shows an example where a token appears abruptly at the beginning of a sentence in the CAMeL Lab version of ANERCorp, resulting in an incoherent or semantically weak context. The lower example shows the original Benajiba version of the same sentence, where the token occurs mid-sentence and is supported by meaningful surrounding context.

This discrepancy likely arises from sentence segmentation or preprocessing errors introduced during dataset standardisation. Such malformed sentence starts produce atypical contextual representations, yet the model assigns labels with high confidence, consistent with the behavioural signature observed for Region A: low representation separability, low uncertainty, and confident misclassification.

Figure 41: Sentence-start misalignment example. **Top**: CAMeL Lab version with an abruptly placed token at sentence start. **Bottom**: Original Benajiba version where the token appears in a semantically coherent context.

A.14 Comparison with Visual Analytics Tools

DeformAR differs from existing visual analytics systems in three key ways, as summarized in Tables 5 and 3.

Analytical focus. T3-Vis analyses training dynamics—tracking how attention heads evolve and which parameters matter for pruning decisions. LIT and InterpreT analyse model behaviour at inference time through counterfactual generation and layer-wise attention inspection. In contrast, DeformAR focuses on post-training evaluation, diagnosing why errors occur by decomposing the system into interacting components (data and model) and tracing how their interactions produce specific error patterns. While all three tools support attention visualization, DeformAR uniquely combines this with span-level error analysis and behavioural metrics to explain prediction failures rather than simply inspect model internals (Table 3).

Task specificity. LIT, InterpreT, and T3-Vis are task-agnostic tools designed for broad applicability, offering generic span inspection at best. DeformAR is purpose-built for sequence labelling, enabling analysis of NER-specific phenomena (span boundary errors, BIO violations, exclusion patterns) that general tools cannot surface. For instance, while existing tools can highlight attention patterns or generate counterfactuals for any task, they cannot distinguish between boundary errors (incorrect span edges) and type errors (wrong entity label), nor can they identify systematic exclusion patterns where entities are consistently predicted as non-entities. These capabilities require task-specific error categorization integrated directly into the evaluation pipeline. While the technical implementation is task-specific, the underlying conceptual framework—component decomposition and interaction analysis—is task-agnostic and can be extended to other tasks by tracking different subcomponents and adapting the metrics and error categorizations accordingly.

Component decomposition. DeformAR explicitly models interactions between data subcomponents (vocabulary, annotations, tokenization) and model subcomponents (representations, output layer). This enables isolation of tokenization effects, annotation artifacts, and representation-output misalignment—capabilities absent in existing tools. For example, DeformAR can trace how WordPiece tokenization introduces ambiguity by causing distinct words to share first subwords, quantify how this affects annotation consistency, and visualize how the resulting confusion manifests in both representation space and prediction behaviour. Similarly, it can identify cases where representations encode correct semantic content (placing a token near entity clusters) while the output layer predicts the wrong label due to memorizing noisy training distributions. Existing tools focus either on training dynamics (T3-Vis) or inference-time exploration (LIT/InterpreT), but do not systematically decompose errors into data-side versus model-side contributions or trace their interactions across the pipeline.

Table 5 provides a multi-dimensional compar-

Capability	T3-Vis	LIT/InterpreT	DeformAR
Span-level errors			✓
BIO violation detection			✓
Tokenization impact			✓
Training dynamics	✓		
Counterfactuals		✓	
Attention visualization	✓	✓	✓
Data-model interaction			✓

Table 3: Capability matrix showing which diagnostic features are supported by each tool.

Tool	Primary Use Case
T3-Vis	Track which attention heads matter during training; identify parameters for pruning
LIT/InterpreT	Generate counterfactuals; inspect layer-wise attention for specific predictions
DeformAR	Diagnose why NER errors occur; isolate data vs. model failures; identify annotation issues

Table 4: Primary use cases for visual analytics tools.

ison across temporal focus, analytical objectives, and unique features. Table 3 shows capability-level differences, highlighting that DeformAR uniquely supports span-level error analysis, BIO violation detection, tokenization impact assessment, and data-model interaction tracing—features essential for diagnostic evaluation of sequence labelling systems. Table 4 summarizes the primary use cases for each tool.

Aspect	T3-Vis	LIT/InterpreT	DeformAR
When	During training	After training (inference)	After training (evaluation)
What	Attention evolution, parameter importance	Prediction explanations, counterfactuals	Error diagnosis, component interactions
NER support	Generic (any task)	Generic span inspection	Span errors, BIO, boundary analysis
Unique to tool	Training trajectory tracking	What-if scenario generation	Data-model-error decomposition

Table 5: Multi-dimensional comparison of visual analytics tools across temporal focus, analytical capabilities, task-specific support, and unique features.