Cross-Domain Adverse Drug Event Recognition Using CRF and Transformer-Based Sequence Labeling Models

Cordell Stonecipher
Department of Computer Science and Engineering
Oakland University
Rochester, MI
Email: stonecipher@oakland.edu

***Abstract*—Adverse Drug Event (ADE) extraction plays a crucial role in pharmacovigilance by identifying harmful side effects of medications mentioned in clinical notes, social media, and patient-generated text. Despite significant progress in biomedical natural language processing, ADE extraction systems suffer from poor cross-domain generalization, largely due to variation in linguistic style, annotation schema, and noise. In this work, we compare classical and transformer-based models across two contrasting ADE datasets: the structured and relatively clean Kaggle ADE corpus [9], and the noisy, patient-generated CADEC corpus derived from online health forums [8].**

A substantial component of our contribution is the development of a unified preprocessing pipeline, including converting Kaggle's Excel dataset into CoNLL format, normalizing CADEC's SNOMED CT–derived labels into a simplified BIO ADE schema, enforcing BIO sequence consistency, aligning token-level labels under subword tokenization, and implementing subsampling methods to allow efficient experimentation on commodity hardware. We evaluate a CRF baseline alongside DistilBERT, BioBERT, PubMedBERT, and MiniLM in both in-domain and cross-domain settings. In-domain performance on Kaggle reaches 1.0 F1 for all models, reflecting dataset simplicity and label balance. In contrast, CADEC presents a significantly more challenging scenario: MiniLM achieves the highest CADEC in-domain F1 score (0.217), outperforming both the CRF and larger biomedical transformers.

Cross-domain evaluation highlights severe performance degradation, with CADEC$\rightarrow$Kaggle transfer achieving moderate success (PubMedBERT F1 ≈ 0.40), while Kaggle$\rightarrow$CADEC performance remains extremely low across all models. Span-level results are complemented with token-level error analysis on ADE and non-ADE tokens, which helps disentangle whether models tend to under-predict or over-predict ADE spans when moved across domains.

Our findings underscore two main insights: (1) transformer contextual embeddings substantially outperform CRFs in noisy biomedical text, and (2) domain mismatch and label imbalance are the primary barriers to robust ADE extraction. We conclude by discussing practical lessons learned from building the end-to-end pipeline and outlining concrete directions for future work, including weighted losses, domain-adaptive pretraining, and joint multi-domain training.

I. Introduction

Adverse Drug Events (ADEs) are harmful or unintended effects associated with medication use. Large-scale detection of ADEs from unstructured text—including clinical notes, biomedical literature, and patient-generated content—is essential for pharmacovigilance, post-marketing surveillance, and patient safety. However, ADE mentions are often sparse, context-dependent, and expressed in informal language, making automated extraction challenging.

Modern ADE extraction systems typically frame the task as sequence labeling with BIO tags, where each token is labeled as beginning, inside, or outside an ADE span. This representation fits naturally into the broader literature on Named Entity Recognition (NER), but the ADE setting presents additional difficulties:

- The signal of interest (ADE spans) is extremely rare compared to background text.
- Patients use varied expressions, non-standard spelling, and colloquial phrasing for symptoms.
- Datasets are heterogeneous and often annotated with different label schemes or medical coding systems.

In this project, we focus on two complementary ADE corpora. The Kaggle ADE dataset [9] contains relatively clean, well-structured text with a simple BIO annotation scheme. In contrast, the CADEC corpus [8] originates from patient reviews on the AskaPatient website and was originally annotated with SNOMED CT concept identifiers, reflecting much noisier, real-world language. Together, these datasets provide a realistic scenario for studying domain shift: models trained on clean text may fail dramatically when faced with noisy patient reviews, and vice versa.

Formally, given a sequence of tokens $\mathbf{x} = (x_1, \ldots, x_T)$, the goal is to predict a sequence of labels $\mathbf{y} = (y_1, \ldots, y_T)$, where $y_t \in \{O, B\text{-}ADE, I\text{-}ADE\}$. Training examples are pairs $(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})$ drawn from one domain (Kaggle or CADEC), and we evaluate both in-domain and cross-domain generalization.

We build an end-to-end pipeline that:

1) Harmonizes the label spaces across datasets via a unified BIO ADE schema;
2) Converts the Kaggle Excel format into a CoNLL-style representation compatible with sequence labeling tools;
3) Aligns token-level labels after subword tokenization for transformer models; and

4) Benchmarks multiple models across in-domain and cross-domain configurations.

In addition to span-level F1, we analyze token-level error rates on ADE versus non-ADE tokens. This provides a more fine-grained view of how models behave on minority vs. majority classes and helps explain why cross-domain performance collapses in some settings. Overall, the project is as much about robust data engineering and evaluation design as it is about model comparison.

II. Background and Related Work

Sequence labeling for biomedical text has been studied extensively, beginning with feature-rich linear models such as Conditional Random Fields (CRFs) [1]. CRFs model the conditional distribution over label sequences given an input sequence and can incorporate handcrafted features capturing capitalization, morphology, and local context. While effective on small, clean datasets, CRFs struggle to encode long-range dependencies and subtle contextual cues in noisy text.

More broadly, NER has a long history in NLP; surveys such as Nadeau and Sekine [5] summarize rule-based, feature-based, and early statistical approaches. Neural NER architectures, including BiLSTM-CNN-CRF models [6], significantly improved performance by learning character-level and word-level representations jointly, while still using a CRF layer to enforce span consistency.

The advent of deep contextual embeddings fundamentally changed NER and related tasks. BERT [2] introduced bidirectional transformer encoders trained with masked language modeling and next sentence prediction objectives. By pretraining on large corpora, BERT learns rich representations that can be adapted to downstream tasks with relatively modest amounts of labeled data.

For biomedical NLP, domain-specific variants have been proposed, including BioBERT [3], which is initialized from BERT and further pre-trained on PubMed and PubMed Central, and PubMedBERT [4], which is trained from scratch solely on biomedical literature. ClinicalBERT and related models [7] focus on clinical notes from electronic health record (EHR) systems. These models have improved performance on a variety of biomedical NER and relation extraction tasks. Lighter-weight models such as DistilBERT and MiniLM aim to retain most of BERT's performance at a fraction of the computational cost.

Despite these improvements, cross-domain generalization remains a major challenge. Models fine-tuned on one dataset often perform poorly on another, especially when moving from formal text (e.g., scientific abstracts) to informal, patient-generated content. This is particularly problematic for ADE extraction, where the target domain is often noisy text from forums or social media.

Our work situates itself in this space by providing a careful empirical comparison of a CRF baseline and several transformer models across the Kaggle and CADEC ADE datasets, combined with a detailed description of all preprocessing steps required to make these datasets interoperable. The emphasis on token-level error analysis complements prior work that focuses primarily on span-level metrics.

III. Data Preparation

Much of the engineering effort in this project went into preparing heterogeneous datasets for consistent training and evaluation. We emphasize these steps because, in practice, they often dominate the total project time and directly influence model quality.

A. Datasets

We use two publicly available ADE corpora:

1) Kaggle ADE Dataset: The Kaggle ADE dataset [9] is released as an Excel file containing token-level annotations. Each row includes a sentence identifier, the token, and a label. The original label set includes ADE mentions and related concepts; for this project we focus on ADE span labels and map all drug-name labels to the outside (O) class to simplify the task to ADE-only extraction. This focuses our analysis on symptom-level ADE detection rather than joint drug–ADE extraction.

2) CADEC Corpus: The CADEC corpus [8] is a collection of user-generated reviews from an online patient forum. Each review is associated with annotations linking spans of text to SNOMED CT concept identifiers, many of which correspond to symptoms or adverse effects. CADEC is considerably noisier than Kaggle: sentences are longer, grammar is informal, and there is substantial spelling variation. Reviews may also contain multiple drugs and conditions in a single narrative, which makes boundary detection more difficult.

B. Kaggle Excel-to-CoNLL Conversion

The Kaggle dataset is originally provided in a spreadsheet-style format. For sequence labeling experiments, a CoNLL-style format (one token per line with labels, sentences separated by blank lines) is far more convenient. We implemented a converter with the following steps:

1) Group rows by sentence ID to reconstruct sentence boundaries.
2) Sort rows within each sentence by token position, if necessary.
3) Remove or repair rows with missing tokens or labels, defaulting missing labels to O when appropriate.
4) Guarantee that every sentence ends with a newline and that there are no empty sentences.
5) Write each sentence as a block of `token label` lines, separated by a blank line.

In addition, we run a sanity check that ensures the resulting BIO sequences are legal: we forbid transitions directly from O to I-ADE (an I tag must be preceded by a B tag of the same type) and correct such cases to B-ADE. This prevents downstream models from learning inconsistent label patterns that originated from annotation or conversion artifacts.

Figure 1 shows where this conversion fits in the overall pipeline from raw data to model-ready features.

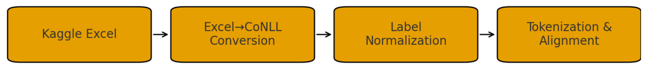

Fig. 1. High-level data pipeline from Kaggle Excel to CoNLL, followed by label normalization and tokenization/label alignment.

C. Label Normalization for CADEC

CADEC uses labels of the form `B-22298006`, `I-3723001`, etc., where the numeric suffix is a SNOMED CT concept identifier. While clinically informative, these labels are too sparse and fine-grained for training a robust sequence labeler with the amount of data available. Moreover, they are incompatible with the Kaggle label space.

To harmonize the datasets, we collapse all CADEC concept labels into generic ADE labels:

$$\text{LABELS} = \{O, B\text{-}ADE, I\text{-}ADE\}.$$

The mapping is:

- Any CADEC label starting with `B-` $\rightarrow$ `B-ADE`.
- Any CADEC label starting with `I-` $\rightarrow$ `I-ADE`.
- All other tokens (including those originally unlabeled) $\rightarrow$ `O`.

We apply a similar simplification to the Kaggle dataset by mapping all non-ADE entity labels (e.g., drug names) to O. After this normalization, both corpora share the same label inventory, and we verified that no labels fall outside `['O', 'B-ADE', 'I-ADE']`.

Although this mapping discards medically rich information, it greatly reduces label sparsity and enables a clean cross-domain evaluation protocol in which models are always predicting the same three labels, regardless of training domain.

D. Subsampling and Dataset Statistics

To make experimentation feasible on a single GPU, we subsampled:

- 2000 Kaggle training sentences,
- 2000 Kaggle test sentences,
- 2000 CADEC training sentences, and
- 1878 CADEC test sentences.

The resulting label distributions are shown in Table I. The counts reveal that CADEC is extremely imbalanced: ADE tokens (B-ADE and I-ADE) are very rare compared to O tokens, especially when compared with Kaggle.

Figure 2 visualizes these distributions. Kaggle has thousands of ADE tokens in both train and test, while CADEC has only hundreds. This heavily constrains how much a model can learn about ADE boundaries from CADEC alone and also makes metrics on CADEC more volatile, since a small number of token-level mistakes can have a large relative impact on span-level scores.

TABLE I
DATASET STATISTICS AFTER PREPROCESSING AND SUBSAMPLING.

Split	Sents	O	B-ADE	I-ADE
Kaggle train	2000	25028	2000	2551
Kaggle test	2000	24967	2000	2494
CADEC train	2000	32576	114	117
CADEC test	1878	30454	131	133

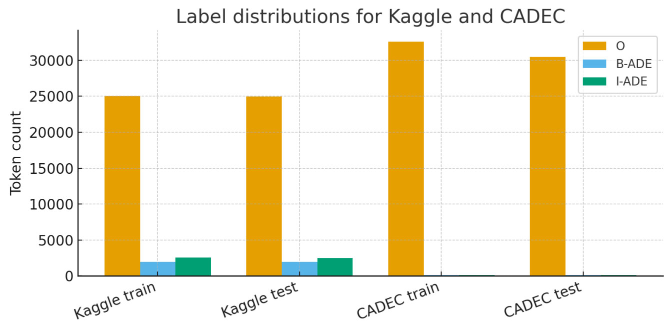

Fig. 2. Label distributions for Kaggle and CADEC splits, derived directly from the counts in Table I.

E. Tokenization and Label Alignment

Transformer models operate on subword units rather than raw tokens. This requires aligning word-level labels with subword tokens. We follow a common approach:

1) Perform WordPiece tokenization on each sentence.
2) For each original word:
   - Assign the original label (e.g., B-ADE) to the first subword.
   - Assign a special ignore index (e.g., `-100`) to all subsequent subwords so they do not contribute to the loss.

As an example:

```
Input word:  "headache"
Tokenized:   "head" "##ache"
Labels:      [B-ADE] [-100]
```

Figure 3 shows an example sentence and how words, tokens, and labels are aligned. In addition to the ignore index, we ensure that the special tokens added by the transformer tokenizer (e.g., `[CLS]`, `[SEP]`) are always assigned `-100` so that they never influence the loss or evaluation metrics.

F. Data Validation

We validated the pipeline through:

- Frequency analysis of label distributions (Table I);
- Random inspection of raw sentences and their CoNLL representations;
- Checks ensuring that ignored subwords never contribute to the training loss; and
- Consistency checks that BIO sequences remain valid after all transformations.

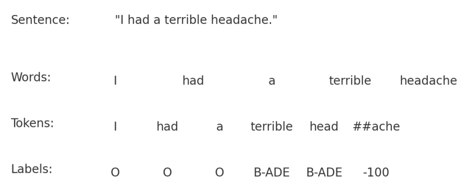

Fig. 3. Illustration of tokenization and label alignment: words are split into subwords, and labels are aligned with the first subword while ignoring subsequent pieces during training.

These checks helped catch early bugs such as misaligned sentence boundaries, incorrect label mappings, or off-by-one errors in tokenization.

IV. Models

A. CRF Baseline

The CRF baseline is a linear-chain Conditional Random Field [1]. Given an input sequence $\mathbf{x}$ and label sequence $\mathbf{y}$, the model defines:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left(\sum_t A_{y_t, y_{t-1}} + P_{t,y_t}\right),$$

where $A_{i,j}$ are transition scores between labels i and j, $P_{t,i}$ are emission scores for label i at position t, and $Z(\mathbf{x})$ is the partition function. We use simple word-level features (token identity, capitalization patterns, and local context).

In practice, emissions $P_{t,i}$ are computed as linear functions of a feature vector extracted from token x_t and its neighbors. Decoding is performed with the Viterbi algorithm, which finds the highest-scoring label sequence under the model. CRFs serve as a useful baseline because they embody the classic feature-based paradigm and make no use of pretraining. Their performance gap relative to transformer models thus reflects the added value of contextual embeddings and large-scale pretraining.

B. Transformer Models

We evaluate four transformer-based token classifiers:

- **DistilBERT**: A distilled version of BERT that retains most of the performance while being smaller and faster.
- **BioBERT** [3]: Initialized from BERT and further pretrained on large biomedical text collections.
- **PubMedBERT** [4]: Trained from scratch solely on PubMed abstracts and articles.
- **MiniLM**: A compact transformer model distilled from larger teacher models using self-attention distillation.

For each model, we attach a linear token classification head that maps the hidden state h_t at position t to label logits:

$$\mathbf{o}_t = W h_t + b, \qquad p(y_t = k \mid \mathbf{x}) = \frac{\exp(o_{t,k})}{\sum_{k'} \exp(o_{t,k'})},$$

where W and b are learned parameters. The classification head is jointly trained with the underlying encoder during fine-tuning.

From an architectural perspective, these models differ primarily in pretraining corpus, number of layers, and hidden dimensionality. BioBERT and PubMedBERT bring specialized biomedical knowledge, whereas DistilBERT and MiniLM prioritize efficiency. Comparing them on the same ADE tasks sheds light on how much domain-specific pretraining matters relative to model size in noisy settings.

V. Training Setup

All transformer models are fine-tuned using HuggingFace Transformers with:

- Learning rate: 5×10^{-5},
- Batch size: 32,
- Epochs: 10,
- Maximum sequence length: 128,
- Dropout: 0.1,
- Optimizer: AdamW.

The loss is standard token-level cross-entropy computed only over tokens whose labels are not `-100`, so that subword fragments and special tokens do not contribute to the objective:

$$\mathcal{L} = -\frac{1}{N} \sum_{(t,i)\,:\, y_t^{(i)} \neq -100} \log p\big(y_t^{(i)} \mid \mathbf{x}^{(i)}\big),$$

where the sum ranges over non-ignored positions in the training batch.

We trained for 10 epochs for all transformer models, monitoring the training loss to ensure that optimization converged. The CRF baseline is trained with standard log-likelihood maximization using L-BFGS optimization. We do not heavily tune CRF-specific hyperparameters, as the focus of the study is on relative trends rather than absolute optimization of the baseline.

All experiments were run on a single GPU, and practical considerations such as constrained VRAM influenced our decision to use DistilBERT and MiniLM in addition to larger biomedical models. We also disabled unnecessary features (e.g., multiple data loader workers) to avoid known issues on Windows-based environments and to keep the overall pipeline simple and reproducible.

VI. Evaluation Methodology

A. Span-Level Metrics

We evaluate ADE extraction primarily at the span level. A predicted span is considered correct only if both its boundaries and label match a gold span exactly. For each experiment, we compute:

- **Precision**: $P = \frac{\text{\# correct predicted spans}}{\text{\# predicted spans}}$,
- **Recall**: $R = \frac{\text{\# correct predicted spans}}{\text{\# gold spans}}$,
- **F1**: $F_1 = \frac{2PR}{P+R}$.

These metrics reflect practical ADE extraction requirements, where partial overlaps may be less useful for downstream pharmacovigilance tasks than perfectly recovered spans.

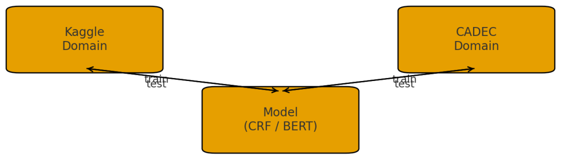

Fig. 4. Schematic of cross-domain experiments. Arrows represent training and testing directions between Kaggle and CADEC domains for each model.

TABLE II
TOKEN-LEVEL ERROR RATES ON CADEC TEST SET. ALL MODELS SHARE 264 ADE TOKENS (B-ADE/I-ADE) AND 30,454 O TOKENS.

Model (train)	**err_ADE**	**err_O**
CRF (kaggle)	0.012516	0.008562
CRF (cadec)	0.013834	0.008459
DistilBERT (cadec)	0.015810	0.010762
PubMedBERT (cadec)	0.015810	0.010556
MiniLM (cadec)	0.017128	0.011105
BioBERT (cadec)	0.018445	0.012715
DistilBERT (kaggle)	0.069829	0.095897
BioBERT (kaggle)	0.079051	0.087398
PubMedBERT (kaggle)	0.121212	0.143812
MiniLM (kaggle)	0.186430	0.195565

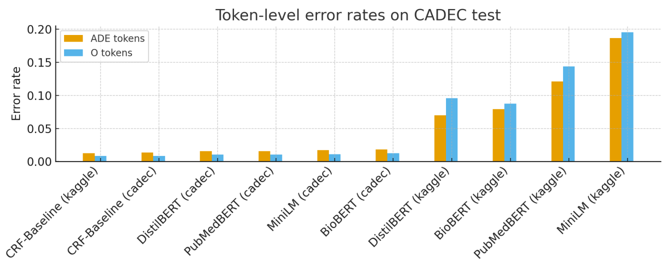

Fig. 5. Token-level error rates on CADEC test set for different models and training domains, derived from Table II.

B. Token-Level Error Rates

Span-level metrics can obscure how models behave on majority vs. minority classes. To gain additional insight, we compute token-level error rates for:

- ADE tokens: positions labeled B-ADE or I-ADE;
- Non-ADE tokens: positions labeled O.

Let T_{ADE} denote the number of ADE tokens and T_O the number of O tokens in the test set. Let M_{ADE} and M_O denote the number of misclassified ADE and O tokens, respectively. We then define

$$\text{err\_ADE} = \frac{M_{\text{ADE}}}{T_{\text{ADE}}}, \qquad \text{err\_O} = \frac{M_O}{T_O}.$$

For CADEC test, we have $T_{\text{ADE}} = 264$ and $T_O = 30{,}454$ (Table I). For Kaggle test, the total number of tokens is 29,461, with ADE and O tokens as in Table I. The error rates reported in our tables use these denominators and the misclassification rates computed by our analysis script.

C. Experimental Settings

We consider three experimental settings:

1) **Kaggle in-domain**: Train and test on Kaggle.
2) **CADEC in-domain**: Train and test on CADEC.
3) **Cross-domain**: Train on one dataset and evaluate on the other.

Figure 4 provides a high-level view of the cross-domain configuration, where arrows indicate possible train-test directions between Kaggle and CADEC. In each configuration, the label inventory is identical, so differences in performance can be attributed to shifts in text style, span distribution, and noise rather than label incompatibility.

VII. RESULTS

A. In-Domain Span-Level Results

On Kaggle, all models—including the CRF—achieve F1 scores of 1.0. This reflects the relative simplicity of the dataset: the label distribution is balanced (Table I), the language is clean, and ADE spans follow consistent patterns. Once label normalization is applied and a reasonable amount of supervision is provided, even a CRF baseline can memorize these patterns effectively.

On CADEC, the story is very different. The CRF baseline achieves substantially lower F1, and even the best transformer model (MiniLM) reaches only 0.217 F1 in-domain. BioBERT, DistilBERT, and PubMedBERT exhibit similar but slightly lower performance. These results illustrate how challenging noisy, user-generated text can be, particularly when ADE spans are rare and often embedded in long, multi-sentence narratives.

B. Token-Level Error Rates on CADEC

Table II reports token-level error rates on the CADEC test set. All models share the same token counts: 264 ADE tokens (B-ADE and I-ADE) and 30,454 O tokens.

Figure 5 visualizes these values. Models trained directly on CADEC maintain relatively low error rates (on the order of 1–2% for ADE tokens and under 1.5% for O tokens). In contrast, models trained only on Kaggle exhibit dramatically higher err_ADE and err_O on CADEC, confirming the severity of the domain shift.

C. Token-Level Error Rates on Kaggle

Table III shows token-level error rates on the Kaggle test set. Models trained on Kaggle achieve zero error on ADE tokens and zero error on O tokens, consistent with 1.0 F1. Models trained only on CADEC, however, achieve err_ADE = 0 but mislabel a substantial fraction (10–22%) of O tokens.

Figure 6 plots these error rates. Cross-domain models trained on CADEC still achieve err_ADE = 0 on Kaggle but mislabel 10–22% of O tokens. This indicates that they tend to over-predict ADE labels in the cleaner Kaggle domain, which is consistent with being trained on noisier data where ADE spans are expressed in more varied ways.

TABLE III
TOKEN-LEVEL ERROR RATES ON KAGGLE TEST SET. ALL MODELS SHARE 29,461 TOTAL TOKENS IN THE TEST SET; THE ADE ERROR RATE IS ZERO IN ALL CONFIGURATIONS.

Model (train)	err_ADE	err_O
CRF (kaggle)	0.000000	0.000000
DistilBERT (kaggle)	0.000000	0.000000
MiniLM (kaggle)	0.000000	0.000000
BioBERT (kaggle)	0.000000	0.000000
PubMedBERT (kaggle)	0.000000	0.000000
CRF (cadec)	0.000000	0.185058
DistilBERT (cadec)	0.000000	0.218390
MiniLM (cadec)	0.000000	0.223821
BioBERT (cadec)	0.000000	0.199823
PubMedBERT (cadec)	0.000000	0.105020

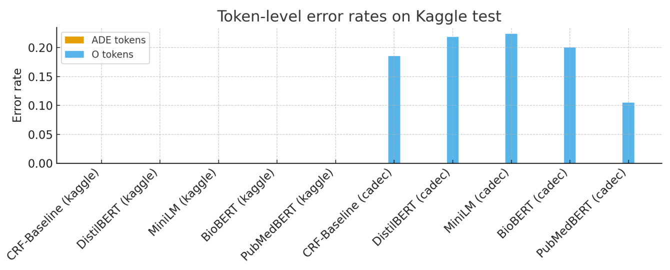

Fig. 6. Token-level error rates on Kaggle test set for models trained on Kaggle vs. CADEC, derived from Table III.

VIII. DISCUSSION

The combined span-level and token-level results yield several insights:

A. Transformers vs. CRF

On Kaggle, both the CRF and transformer models achieve perfect or near-perfect performance, suggesting that model capacity and pretraining are less critical when the domain is clean and the label distribution is relatively balanced. On CADEC, however, transformers outperform the CRF baseline in span-level F1, even though their token-level error rates on ADE tokens are within roughly one percentage point of the CRF (Table II). This suggests that contextual embeddings help the transformers better identify the boundaries of ADE spans and make more coherent span-level predictions, even when the overall number of token-level mistakes is similar.

B. Impact of Label Imbalance

Figure 2 and Table I highlight how heavily imbalanced CADEC is compared to Kaggle. With only 264 ADE tokens in the CADEC test set, small changes in err_ADE can produce large changes in span-level F1. This explains why all models achieve relatively low ADE F1 on CADEC despite modest-looking token-level error rates: most errors are concentrated on a tiny minority class, and a single missed span may correspond to many consecutive ADE tokens.

C. Asymmetric Domain Shift

The token-level error tables also reveal an asymmetry in domain transfer. Models trained on CADEC exhibit moderate error on Kaggle O tokens (Table III), whereas models trained on Kaggle show severe increases in both err_ADE and err_O on CADEC (Table II). Intuitively, exposure to noisier CADEC text may make models more tolerant of variation, allowing them to adapt more easily to cleaner Kaggle sentences. The reverse is not true: models trained only on clean Kaggle data are not prepared for the spelling variation, informal phrasing, and distribution shift present in CADEC.

D. Behavior of MiniLM

MiniLM achieves the best CADEC in-domain F1 while not always having the lowest token-level error rates. One possible explanation is that its smaller capacity and distillation process act as a regularizer, enabling it to better capture the general structure of ADE spans without overfitting to rare patterns. In practice, this makes MiniLM a strong candidate for deployment scenarios where computation must be constrained and latency is a concern.

E. Implications for Real-World ADE Mining

From a practical standpoint, the results suggest that simply fine-tuning a large biomedical transformer on a clean dataset like Kaggle may not be sufficient for real-world ADE surveillance on social media or patient forums. Instead, practitioners should anticipate severe domain shift and consider training directly on noisier data, even if that data is smaller and highly imbalanced. The token-level error analysis underscores that most mistakes arise from majority-class confusion (O vs. ADE) when models are moved across domains, pointing to the need for domain adaptation and imbalance-aware techniques in realistic deployments.

IX. LIMITATIONS

This study has several limitations:

- The datasets are subsampled to reduce training time, which may exclude some rare ADE patterns and limit the diversity of linguistic constructions seen during training.
- The unified label schema collapses rich clinical information (e.g., specific SNOMED CT concepts) into a single ADE label, which is convenient for cross-domain comparison but not sufficient for fine-grained pharmacovigilance.
- We did not explore class weighting, focal loss, or data augmentation techniques that could better handle label imbalance, especially for CADEC.
- All transformer models were trained for 10 epochs with a single set of hyperparameters; it is likely that careful tuning could further improve results, particularly on CADEC.
- We did not apply more advanced domain adaptation approaches such as domain-adaptive pretraining, adversarial training, or multi-task learning, which may substantially improve cross-domain performance.

X. Future Work

Several extensions present themselves naturally:

- Incorporating class-weighted or focal loss to emphasize ADE tokens during training, especially on CADEC.
- Oversampling sentences containing ADE spans or using curriculum learning to present easier examples first, gradually increasing difficulty.
- Training a joint model on both Kaggle and CADEC, possibly with domain-specific adapters or prompts that allow the model to specialize representations for each domain while sharing parameters when possible.
- Applying domain-adaptive pretraining on unlabeled CADEC and Kaggle text to better align representations and reduce covariate shift between clean and noisy biomedical text.
- Adding a CRF layer on top of transformer encoders to refine span boundaries while leveraging contextual embeddings, following the BiLSTM-CNN-CRF paradigm [6].
- Extending the token-level error analysis to include confusion matrices and per-span length breakdowns, to better understand whether models struggle more with short or long ADE mentions.

XI. Conclusion

We presented an end-to-end pipeline and empirical study for cross-domain ADE extraction using CRF and transformer-based models on the Kaggle ADE dataset and the CADEC corpus. After significant data preparation work—including label normalization, Excel-to-CoNLL conversion, and subword label alignment—we benchmarked multiple architectures in both in-domain and cross-domain settings.

Our experiments show that while Kaggle is relatively easy for all models, CADEC remains challenging due to noise and extreme imbalance. Transformers outperform the CRF baseline on CADEC, and MiniLM provides a particularly attractive trade-off between performance and efficiency. Token-level error analysis reveals that cross-domain failures are dominated by domain shift and label imbalance rather than model capacity alone.

These findings highlight the importance of robust preprocessing and domain-aware training strategies for real-world ADE extraction and motivate future work on domain adaptation and imbalance-aware learning. More broadly, the study illustrates how even simple modeling choices can interact in complex ways with annotation schemas, label distributions, and text styles when moving beyond clean benchmark datasets.

Appendix A
Preprocessing Algorithms

For reproducibility, we summarize the main preprocessing steps as high-level pseudocode.

A. Kaggle Excel-to-CoNLL Conversion

```
for each sentence_id in
    sorted(unique(sentence_ids)):
   rows = all_rows_with_this_sentence_id
   sort rows by token_index
   for row in rows:
      token = row.token
      label = row.label
      if label is missing:
         label = "O"
      write(f"{token} {label}")
   write("") # blank line between sentences
```

B. Label Normalization

```
def normalize_label(label):
    if label.startswith("B-"):
        return "B-ADE"
    elif label.startswith("I-"):
        return "I-ADE"
    else:
        return "O"
```

This function is applied to every token-level label in CADEC. For Kaggle, any label corresponding to a drug name or non-ADE concept is mapped to "O" using a similar rule-based function.

Appendix B
Tokenization and Alignment Procedure

The following pseudocode outlines the alignment of word-level labels with subword tokens for transformer models:

```
for sentence, word_labels in dataset:
   word_pieces = []
   aligned_labels = []
   for word, label in zip(sentence,
       word_labels):
      pieces = tokenizer.tokenize(word)
      for i, piece in enumerate(pieces):
         word_pieces.append(piece)
         if i == 0:
            aligned_labels.append(label)
         else:
            aligned_labels.append(-100) #
                ignore index
   # Add special tokens
   word_pieces = [CLS] + word_pieces + [SEP]
   aligned_labels = [-100] + aligned_labels +
       [-100]
```

This algorithm ensures that the loss is computed only on the first subword of each original token while propagating contextual information through all subwords in the encoder.

Appendix C
CRF Feature Template

For completeness, we sketch a simple set of features used in the CRF baseline:

- Current token (lowercased).
- Previous and next tokens (lowercased).
- Binary flags for capitalization patterns (all caps, first letter capitalized, all lowercase).
- Binary flag indicating whether the token contains digits.

- Suffixes and prefixes of length 2–3, to capture common morphemes.

These features are computed at every position in the input sequence and concatenated into a fixed-size vector that feeds into the CRF emission layer.

References

[1] J. Lafferty, A. McCallum, and F. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data," in *Proc. ICML*, 2001.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proc. NAACL*, 2019.

[3] J. Lee, W. Yoon, S. Kim, et al., "BioBERT: a pre-trained biomedical language representation model for biomedical text mining," *Bioinformatics*, vol. 36, no. 4, pp. 1234–1240, 2020.

[4] Y. Gu, R. Tinn, H. Cheng, et al., "Domain-Specific Language Model Pretraining for Biomedical Natural Language Processing," *Transactions of the ACL*, vol. 9, pp. 1–15, 2021.

[5] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *Lingvisticae Investigationes*, vol. 30, no. 1, pp. 3–26, 2007.

[6] X. Ma and E. Hovy, "End-to-end Sequence Labeling via BiLSTM-CNNs-CRF," in *Proc. ACL*, 2016.

[7] E. Alsentzer, J. Murphy, W. Boag, et al., "Publicly Available Clinical BERT Embeddings," in *Proc. NAACL Clinical NLP Workshop*, 2019.

[8] CSIRO, "CSIRO Adverse Drug Event Corpus (CADEC)," CSIRO Data Collection, 2015. [Online]. Available: https://data.csiro.au/collection/csiro:10948

[9] M. Ashraf, "Adverse Drug Event (ADE) Prediction and Analysis," Kaggle, 2023. [Online]. Available: https://www.kaggle.com/datasets/mohammedashraf000/adverse-drug-event-ade-prediction-and-analysis