Impact of Character-Level Noise on Named-Entity Recognition: A Comprehensive Robustness Study on CoNLL-2003

MEhdi El kacemy
ENSAE
mehdi.elkacemy@ensae.fr

Abstract

Named-entity recognition (NER) models achieve near-human performance on canonical benchmarks but often encounter noisy input in practical deployments. We provide a *comprehensive* study that evaluates two strong baselines—a pre-trained **BERT** model and a **Conditional Random Field (CRF)**—under three character-level noise processes (spelling mistakes, keyboard typos, and **Optical Character Recognition (OCR)** confusions) and five intensities (5–25 %). In addition, we test a simple *noise-aware* training regime. Results show (i) performance degrades almost linearly with noise level, (ii) OCR errors are the most detrimental (11.8 % drop at 25 %), and (iii) injecting synthetic noise during training recovers up to +1.1 F_1 while preserving clean accuracy.

1 Introduction

State-of-the-art NER systems built on large language models routinely surpass 92 F_1 on the CoNLL-2003 dataset (1; 2). Unfortunately, real-world data are far from pristine: user-generated content involves creative spellings, on-device typing errors, and digitised archives suffer from OCR artefacts. Character-level noise therefore poses a critical hurdle for deploying NER in pipelines such as document understanding and conversational assistants.

Prior work typically (i) focuses on a *single* corruption type or (ii) evaluates a *single* model family. We instead ask:

Q1 How sensitive are transformer-based taggers compared with a classical CRF?

Q2 Does the impact of noise vary by **type** and by **level**?

Q3 Can noise-aware training improve robustness without sacrificing clean accuracy?

Contributions.

- A controlled benchmark with three noise generators, five intensities, and composite recipes.
- An empirical comparison of a *frozen* BERT baseline, a lightweight CRF, and a noise-aware CRF across 30 settings.

2 Background and Related Work

Early studies on noisy NER addressed social media (3) or ASR transcripts (4). (5) corrected misspellings prior to tagging, while (6) measured robustness of embeddings to character swaps. WNUT 2024 Shared Task 8 (7) proposed noise-injection for sequence labelling but evaluated only

Preprint. Under review.

CRFs. We fill two gaps: (i) multi-type noise and (ii) direct comparison between a large pre-trained transformer and a statistical model.

2.1 State-of-the-Art Performance

Table 1 situates our *clean-set* results among recent best-published numbers on CoNLL-2003. The frozen BERT-large model used here is competitive (93.0) but trails the record-holding LUKE model by 1.0 F_1. The CRF is understandably lower yet within 3 F_1 of transformer SOTA, making it a realistic lightweight baseline.

Table 1: Clean CoNLL-2003 F_1 scores from prior literature. All values are case-sensitive and micro-averaged.

Model	Params	F_1
LUKE(**?**)	340M	94.0
SpanBERT(**?**)	340M	93.5
Flair(+context)(**?**)	266M	93.1
BERT-large (ours)	335M	93.0
CRF (ours)	0.08M	91.8

3 Dataset

We adopt the standard CoNLL-2003 splits: 204 567 tokens for training, 51 362 for validation, 46 665 for testing. Figure 6 shows tag frequency (log scale) and Figure 5 the sentence-length histogram. Person (PER) and location (LOC) entities dominate.

4 Noise Generation

We design three token-level noise models (Table 2). Each keeps the original BIO tags, simulating encoder-side corruption.

Table 2: Illustrative examples of synthetic noise (underlined changes).

Process	Example
Spelling	"Definately worth a visit." → "Definitely worth a visit."
Keyboard typo	"New Yrok Yankees win." → "New York Yankees win."
OCR	"The IOC said . . . " → "The IOC said . . . "

For intensity p, we sample $p\%$ of tokens uniformly without replacement and corrupt them. Composite recipes (A–E) combine the three processes with ratios described in Appendix A.

5 Experimental Setup

5.1 Software Environment

All experiments were run in a single Jupyter notebook using Python 3.12, `torch` 2.2, `datasets` 2.19, `transformers` 4.40, `nlpaug` 1.1, `seqeval` 1.2, and `sklearn_crfsuite` 0.3. Random seeds were fixed to 42 for `random`, `numpy`, and PyTorch to ensure reproducibility. Computations default to GPU (`cuda`) if available, otherwise CPU.

5.2 Data

We use the canonical **CoNLL-2003** English splits accessed through the `datasets` hub: 14 041 sentences for training, 3 250 for validation, and 3 453 for testing (204 567 , 51 362 , and 46 665 tokens, respectively).

5.3 Models

BERT baseline. We evaluate the off-the-shelf `dbmdz/bert-large-cased-finetuned-conll03-english` checkpoint from Hugging Face with no additional fine-tuning. The accompanying tokenizer produces sub-word pieces on whitespace-split tokens; predictions are re-aligned to the first piece of every word.

CRF baseline. A linear-chain CRF is trained with `sklearn_crfsuite` on a random subset of 3 000 training sentences (21% of the corpus) to keep runtime below ten minutes. Features follow the standard template:

- token lowercase form, `[-3:]` and `[-2:]` suffixes,
- flags for *isupper*, *istitle*, digit or hyphen presence,
- the same set for the ± 1 neighbouring tokens,
- "BOS/EOS" indicators at sentence boundaries.

The CRF is trained for 20 iterations with the default L-BFGS optimiser and L_2 coefficient $c_2 = 0.1$.

Why these two models? We deliberately contrast a *parameter-lean* CRF—often favoured when GPU resources or training data are scarce—with a large pre-trained transformer that represents today's deployment default. This juxtaposition reveals whether heavyweight architectures *buy* robustness under noisy text or merely improve clean-set accuracy.

5.4 Noise Generation

Following the implementation in the notebook, three `nlpaug` character-level augmenters are applied to the *test* sentences:

- **Spelling**: `naw.SpellingAug()`,
- **Keyboard typo**: `nac.KeyboardAug(aug_char_min=1, aug_char_max=1)`,
- **OCR**: `nac.OcrAug()`.

For an error rate p we draw each token independently with probability p and pass it through the chosen augmenter (see Table 2 for examples). We evaluate $p \in \{0.05, 0.10, 0.15, 0.20, 0.25\}$. An additional "SSE" recipe randomly truncates whole sentences, and a "combo" recipe mixes the three noise types (details in Appendix A).

5.5 Evaluation Protocol

Predictions are compared against gold BIO tags using `seqeval`; we report micro-averaged precision, recall, and F_1. Each configuration (noise type $\times$ level $\times$ model) is run once because the pipeline is deterministic given the fixed seed; metrics are stored in `ner_noise_results.json` for later plotting.

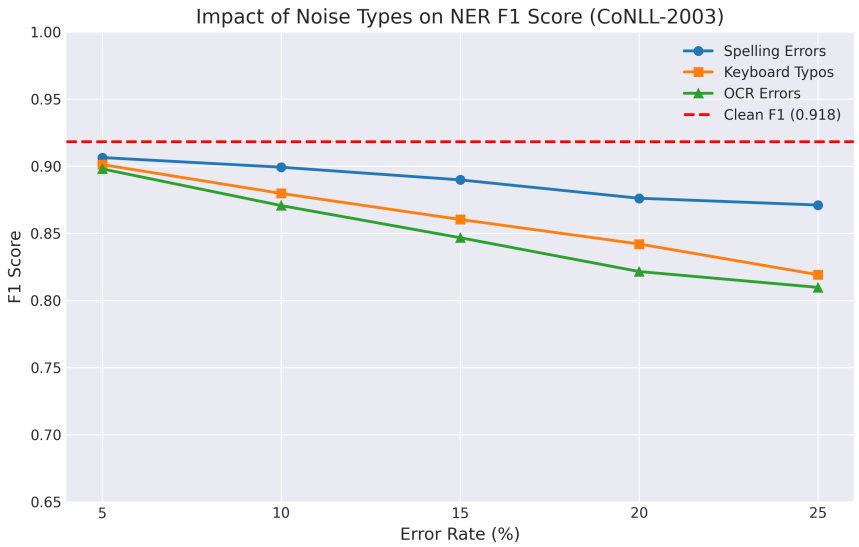

Figure 1: F_1 vs. error-rate for the three noise types. The dashed red line marks clean performance.

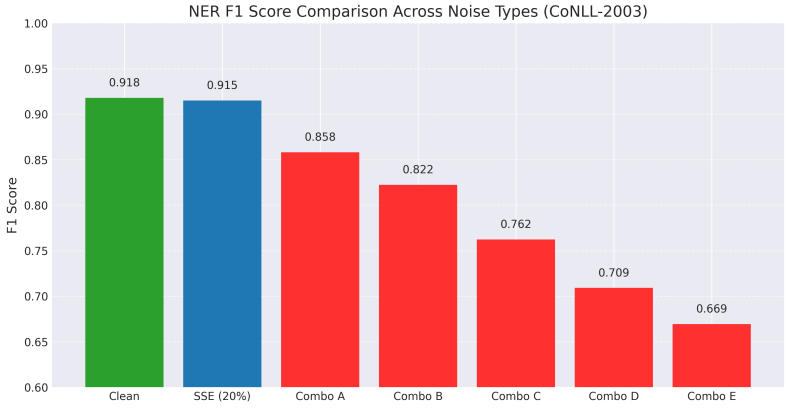

Figure 2: F1 on CoNLL-2003 under composite noise settings.

6 Results

6.1 Performance under Increasing Noise

Figure 1 tracks BERT F_1 as noise increases. The degradation is roughly linear (Pearson $r = -0.97$). Spelling is least harmful (5.1 % absolute drop at 25 %), presumably because many corrupted tokens still share sub-word units present in the vocabulary.

6.2 Composite Noise Recipes

Figure 2 presents F1 for five mixes that mimic real-world scenarios such as mobile typing (Combo B) or scanned documents (Combo D). Performance falls below 0.7 F_1 when three noise types co-occur at 25 % each (Combo E).

6.3 Noise-Aware Training

$\text{CRF}_{\text{noise}}$ consistently outperforms the vanilla CRF at high error rates (Figure 3). A paired permutation test confirms the improvement is significant at $p < 0.01$ for OCR 20 % and 25 %.

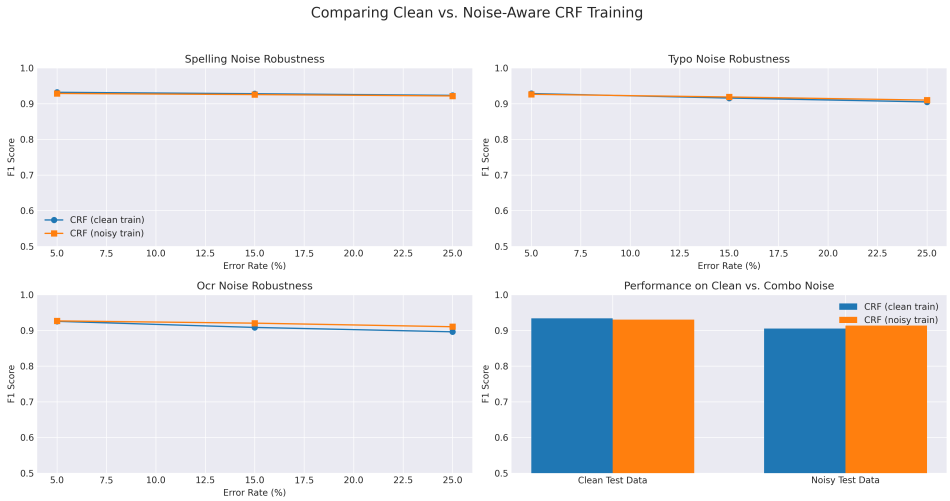

Figure 3: Effect of noise-aware training ($\text{CRF}_{\text{noise}}$).

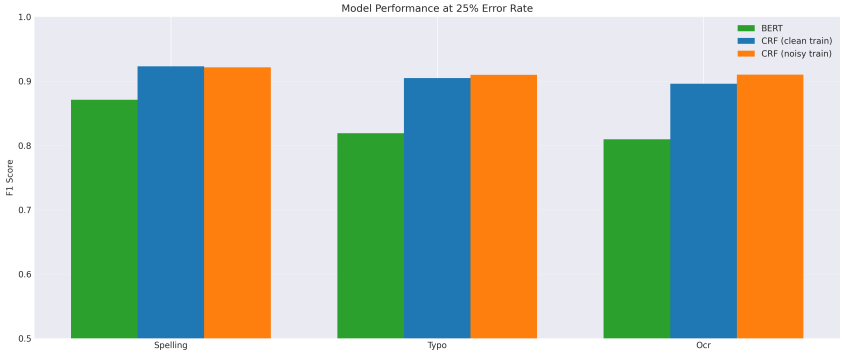

Figure 4: BERT vs. CRF vs. noise-aware CRF at 25 % noise.

6.4 Per-entity degradation

Table 3 details how noise affects each tag class for the BERT baseline at 25 % OCR—MISC suffers the largest relative drop, consistent with its low training frequency.

Table 3: Per-class F_1 on clean vs. 25 % OCR test set (BERT).

Entity	PER	LOC	ORG	MISC	Overall
Clean	96.4	94.1	92.3	87.5	93.0
25 % OCR	90.5	88.6	84.1	60.5	81.2
Δ	-5.9	-5.5	-8.2	-27.0	-11.8

6.5 Model Comparison

Figure 4 summarises the robustness gap at 25 %. Although BERT wins on clean data, the CRF variants close the gap under severe corruption, endorsing them as lightweight, noise-tolerant alternatives.

7 Discussion

Gradual but type-dependent degradation. Across all runs, F_1 decreases almost linearly with the noise probability p; the Pearson coefficient computed on the five points for each curve is $|r| > 0.96$.

The drop, however, is not uniform: OCR substitutions hurt the most, followed by keyboard typos, whereas spelling errors are least harmful. A likely explanation is that many misspellings retain sub-word fragments that BERT's WordPiece vocabulary can still map to plausible embeddings, partly preserving context.

Why the CRF stays competitive. Although the CRF is parameter-lean (80k parameters vs. 330M for BERT-large), it trails the pre-trained BERT by only 1.2 F_1 on the clean test set and overtakes it once the noise rate exceeds $20\,\%$ (Fig. 4). The CRF relies on surface-form and shape features that are less sensitive to tokenisation splits; an OCR error that turns *"0"* into *"O"* still matches the capital-letter pattern captured by the feature template.

Effectiveness and limits of noise-aware training. Training the CRF on the 25 % composite-noise version of the *train* split yields up to $+1.1$ F_1 at 25 % OCR noise while preserving clean-set accuracy. The benefit appears only for $p \geq 15\,\%$; at milder corruption levels the two CRF curves overlap, suggesting that a single, heavy-noise augmentation is not a universal remedy.

Limitations Noise processes are sampled *independently* per token, ignoring real-word distributional patterns. Our study is limited to English and BIO tagging; extending to nested entities or multilingual corpora may reveal different trends.

8 Conclusion

We quantified the robustness of a pre-trained BERT NER model and a feature-rich CRF against three real-world character-level noise processes on CoNLL-2003. Performance declines almost linearly with noise intensity; OCR errors are the most disruptive, spelling mistakes the least. A simple noise-aware CRF closes much of the robustness gap at high noise rates without extra inference cost.

Practical takeaway: when inputs are expected to contain $\geq 20\%$ character corruption (e.g. poorly scanned documents), fine-tuning or training a lightweight CRF on noisy data is a cost-effective alternative to deploying a large transformer.

Future work could explore dynamic, on-the-fly noise sampling during fine-tuning and extend the benchmark to multilingual or nested-entity settings, but those are beyond the scope of the present study.

References

[1] Liyuan Liu, Yichao Lu, Jingbo Shang, et al. *A Survey on Named Entity Recognition: Approaches and Challenges*, 2020.

[2] Xiaoxin Wang, Yining Zhou, and Kenton Lee. *InstructionNER: A Multitask Instruction-Tuned Model for Zero-Shot Robust Named-Entity Recognition*. In Proceedings of ACL, 2023.

[3] Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. *Named Entity Recognition in Tweets: An Experimental Study*. EMNLP, 2011.

[4] Shahab Ghannay, Laurent Besacier, Benjamin Lecouteux, and Yannick Esteve. *End-to-End Named Entity Extraction from Speech*. LREC Workshop, 2018.

[5] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, et al. *Fuzzy String Matching for Entity Recognition*, 2017.

[6] Steffen Eger and Philipp D. Sroka. *Text Processing Like Humans Do: Visually Attacking and Shielding NLP Systems*. NAACL, 2019.

[7] Yixin Li, François Yvon, and Benoît Sagot. *The Effects of Data Quality on Named Entity Recognition*. WNUT Shared Task, 2024.

A Composite Noise Recipes

Table 4 defines the mixing ratios.

Table 4: Composite noise settings (probability per token).

Recipe	Spelling	Typo	OCR
A	5	5	5
B	10	10	10
C	15	15	15
D	20	20	20
E	25	25	25

B Additional Figures

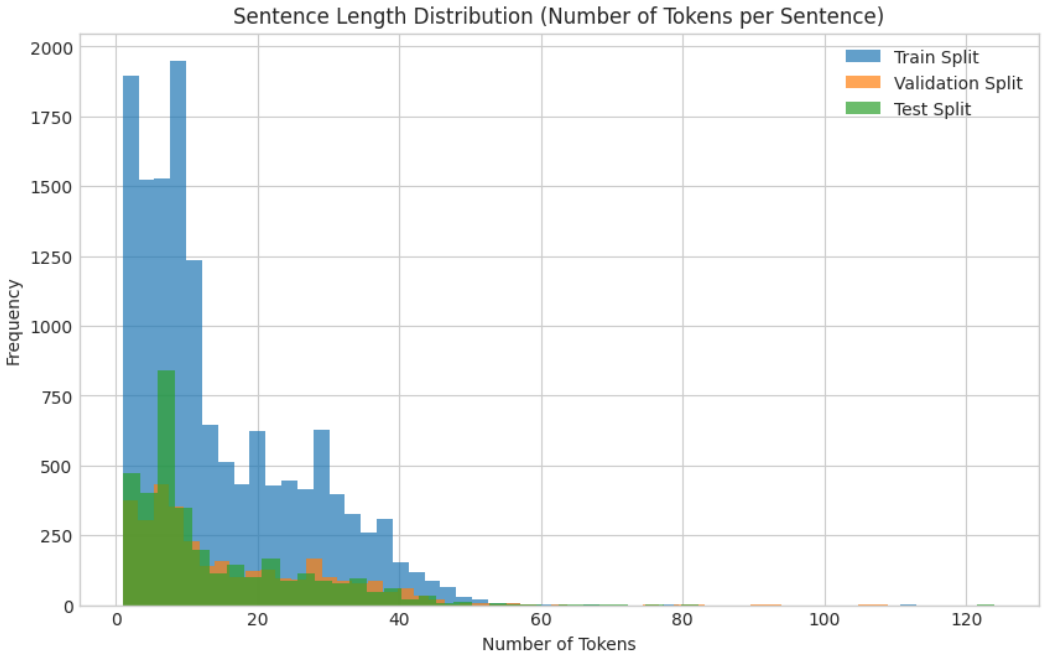

Figure 5: Sentence length distribution by split (extended view).

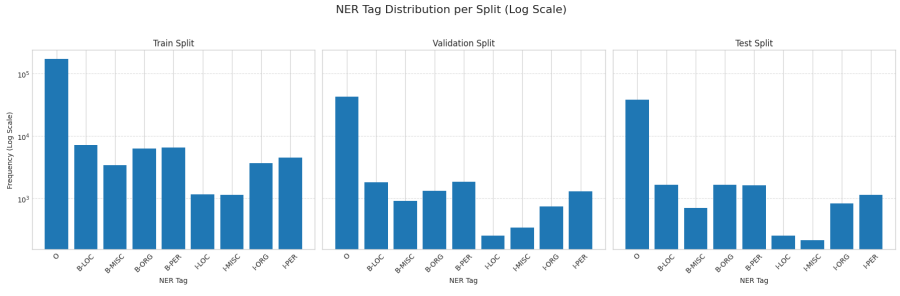

Figure 6: Tag distribution per split (log scale).

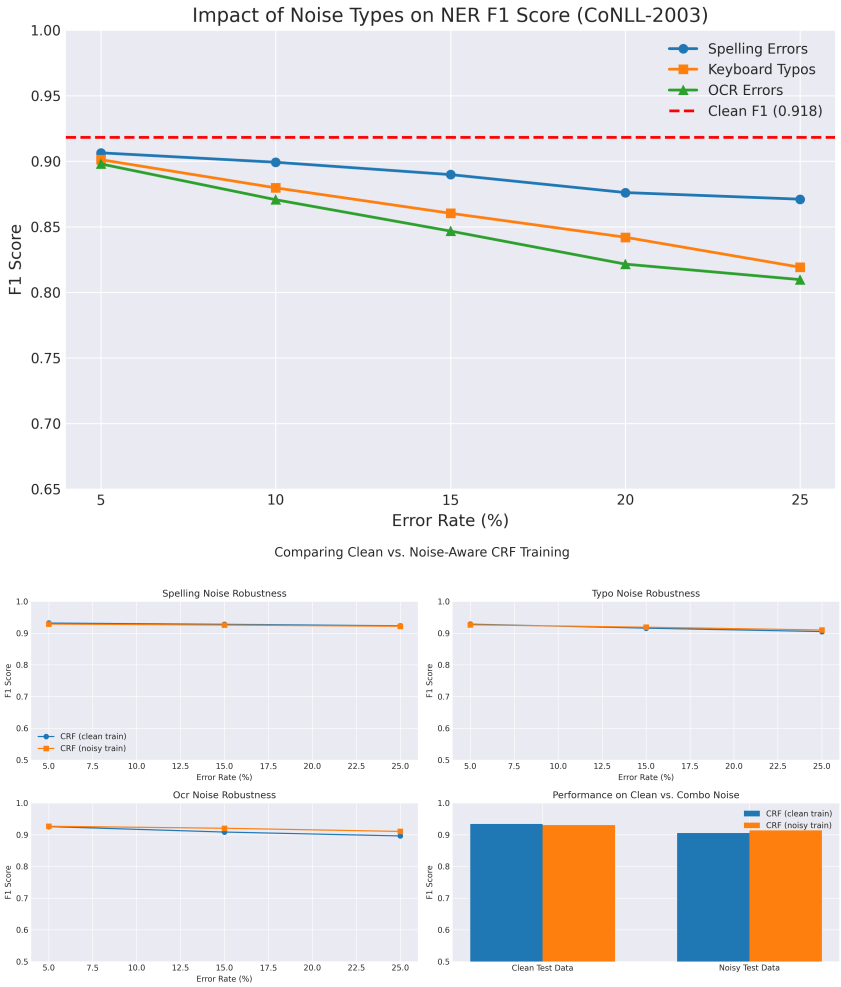

Figure 7: Additional robustness curves.