Vol. 1, No. 1, 2026

Journal of Applied Computational Linguistics

ISSN 2999-0001 (print)
ISSN 2999-0002 (online)
https://jacl.example.org

Table of Contents

Editorial Board ii
Editorial: Why Computational Linguistics Needs Applied Journals 1
Tokenization Drift in Multilingual Pipelines 3
Low-Resource Named Entity Recognition for Regional Dialects 17
Evaluating Machine Translation of Idiomatic Expressions 31
Corpus Construction for Historical Orthography 44
Book Review: Statistical Methods for Text Analysis 58
Call for Papers 60

Editorial Board

Editor-in-Chief
Dr. Helena Marquez, University of Northbridge

Associate Editors
Prof. Tomasz Wiśniewski, Eastern Polytechnic Institute
Dr. Aisha Rahman, Coastal Research Laboratory
Dr. Jonathan Pierce, Meridian Technical College

Advisory Board
Prof. Emeritus Robert Kellogg, Northbridge
Dr. Mei-Ling Chen, Harbor Institute of Technology
Dr. Sofia Andersson, Nordic Language Technology Center
Prof. Diego Alvarado, Southern Cross University

Editorial: Why Computational Linguistics Needs Applied Journals

Dr. Helena Marquez, University of Northbridge

Computational linguistics has long been a field of two cultures. On one side are researchers who build models and publish in fast-moving conference proceedings. On the other are practitioners who deploy those models in production and rarely write anything down. The gap between these two groups is wider than it should be, and it is costing the field real progress.

This journal exists to narrow that gap. We publish work that is both methodologically sound and grounded in a real application, and we give equal weight to negative results and to engineering reports that would not survive peer review elsewhere.

Why another journal?

There are already many venues for computational linguistics. What is missing is a place where a paper can be rejected for being too theoretical. Our reviewers are instructed to ask one question above all others: *could someone actually use this?* If the answer is no, the paper will be returned with suggestions, not accepted.

This is not a criticism of theoretical work. Theory is essential. But theory that never touches data, users, or constraints tends to drift into self-reference, and we want to publish the kind of work that keeps the field honest.

What we publish

We publish full research articles, short reports, tutorials, and reviews of books and software. Research articles should be between 4,000 and 8,000 words. Short reports are capped at 2,000 words and are ideal for negative results, replication studies, and small but useful tools.

Tutorials are a special case. We commission most of them, but we welcome proposals. A good tutorial teaches a reader to do something they could not do before, and it includes code that runs.

Open data and open code

Every empirical paper must release its data and code, or explain clearly why it cannot. "Proprietary" is rarely a sufficient reason. If you are working with sensitive data, we can help you find a way to share a synthetic or anonymized version.

We do not require a specific license, but we strongly prefer permissive ones. Reproducibility is the point of publication, and a paper that cannot be reproduced is only half published.

Timeline

Our target is a first decision within 60 days of submission and publication within 90 days of acceptance. We will not always hit these numbers, but we will tell you when we miss them.

Tokenization Drift in Multilingual Pipelines

Aisha Rahman, Coastal Research Laboratory
Jonathan Pierce, Meridian Technical College

Abstract. Tokenization is the first step in most natural language processing pipelines, and it is also the step most likely to silently change when a model is updated. We describe a class of failures we call tokenization drift, in which a new version of a tokenizer produces different token boundaries on the same input, causing downstream models and evaluation scores to shift without any change to the model itself. We measure drift across 14 tokenizer pairs in 9 languages and show that it can account for more than 3 points of accuracy on a standard named entity recognition benchmark. We propose a lightweight monitoring procedure and release a command-line tool that implements it.

1. Introduction

A tokenizer is rarely the most interesting component of a system, and it is rarely the component that receives the most attention during an upgrade. Teams update tokenizers for good reasons: to add a new vocabulary, to fix a bug, to support a new script. The assumption is that a better tokenizer will produce better downstream results. Sometimes it does. Sometimes it does not, and sometimes it produces different results that nobody notices for months.

We call this tokenization drift. It is the change in token boundaries that occurs when a tokenizer is replaced, retrained, or reconfigured, and it is distinct from the change in model behavior that follows. Drift is measurable on its own, before any model is run, and it is a useful early warning that a pipeline is about to change.

2. Related work

Sennrich et al. (2016) introduced subword tokenization and with it the practice of training tokenizers on large corpora. Kudo and Richardson (2018) released SentencePiece, which made it easy to train and share tokenizers as standalone artifacts. Both papers treat the tokenizer as a fixed preprocessing step.

Work on robustness to tokenization has mostly focused on adversarial inputs (Wallace et al., 2019) and on character-level models that avoid tokenization entirely (Clark et al., 2022). Less attention has been paid to the boring case, where the tokenizer changes between two versions of the same library and the model is left to cope.

3. Measuring drift

Let T_1 and T_2 be two tokenizers and let x be a string. We represent each tokenization as a set of character spans. The drift on x is one minus the Jaccard similarity of those two sets:

$$d(x) = 1 - \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|}$$

where S_1 and S_2 are the span sets produced by T_1 and T_2. The drift on a corpus is the mean of $d(x)$ over all strings. A drift of zero means the tokenizers agree everywhere; a drift of one means they share no boundaries at all.

We also report the fraction of strings whose drift is nonzero, which is easier to interpret when most changes are small.

4. Experiments

We ran 14 tokenizer pairs, drawn from three open-source libraries and their successive releases, over 9 languages. For each pair we tokenized the validation split of a standard named entity recognition benchmark and measured drift at the string and corpus level.

Language	Pairs	Mean drift	Strings changed
English	3	0.012	4.1%
German	2	0.031	9.8%
Turkish	2	0.058	17.3%
Finnish	2	0.064	19.0%
Japanese	2	0.041	12.6%
Arabic	1	0.027	8.4%
Hindi	1	0.022	7.1%
Vietnamese	1	0.019	6.0%

Drift was highest in agglutinative languages, where a single change to a merge rule can affect long words in many places. Drift was lowest in English, where most changes touch only rare vocabulary.

5. Monitoring procedure

We propose running a drift check whenever a tokenizer is updated, before any model is retrained. The procedure is cheap: it requires only the two tokenizers and a sample of text, and it finishes in minutes on a laptop.

```
tokdrift --old tok_v1.model --new tok_v2.model --sample corpus.txt --report drift.html
```

The report lists the strings with the largest drift and the most common boundary changes, which is usually enough to decide whether the update is safe.

6. Conclusion

Tokenization drift is not a problem with any single tokenizer. It is a problem with the assumption that tokenizers are stable. They are not, and pipelines that treat them as such will keep being surprised. Measuring drift is cheap, and we hope it becomes a routine part of tokenizer maintenance.

Low-Resource Named Entity Recognition for Regional Dialects

Tomasz Wiśniewski, Eastern Polytechnic Institute
Aisha Rahman, Coastal Research Laboratory

Abstract. We study named entity recognition for regional dialects that are closely related to a well-resourced standard language but differ enough in vocabulary and spelling that standard models perform poorly. We collect a small annotated corpus for three such dialects and compare four strategies: training on the standard language only, fine-tuning on the dialect, multilingual training, and data augmentation by spelling normalization. Fine-tuning on as few as 2,000 dialect sentences recovers most of the lost accuracy, and normalization helps when the dialect data is scarcer still.

1. Introduction

Most named entity recognition systems are trained on newswire in a standard language, and they work well on newswire in a standard language. They work much less well on anything else. Regional dialects are a common case: the grammar is close enough that a human reader has no trouble, but the vocabulary, spelling, and word order are different enough that a model trained on the standard language misses a large fraction of entities.

We ask how much dialect data is needed to close this gap, and whether the cheapest strategy is also the best one.

2. Related work

Cross-lingual transfer (Pires et al., 2019) and multilingual pretraining (Conneau et al., 2020) have made it possible to build NER systems for languages with little data, by borrowing from related languages. Dialects are a special case of this problem, and they have received less attention than distant languages.

Spelling normalization for historical text (Bollmann, 2019) is a close relative of our normalization step. We borrow its rule-based approach and adapt it to modern dialects.

3. Data

We annotate a corpus of 2,000 sentences for each of three dialects, using the same entity inventory as the standard-language benchmark. Annotation was done by two native speakers per dialect, with a third adjudicating disagreements. Inter-annotator agreement was high enough that we do not report it here.

4. Strategies

We compare four strategies.

1. **Standard only.** Train on the standard-language benchmark and apply the model to dialect text.
2. **Fine-tune.** Start from the standard-language model and continue training on dialect data.
3. **Multilingual.** Train on the standard-language benchmark and dialect data together, with a language tag.
4. **Normalize.** Map dialect spelling to standard spelling with a rule set, then apply the standard-language model.

5. Results

Strategy	Dialect A	Dialect B	Dialect C
Standard only	41.2	38.7	44.5
Fine-tune	78.9	76.4	80.1
Multilingual	74.3	72.0	75.8
Normalize	69.5	66.1	71.2

Fine-tuning on 2,000 sentences recovers most of the lost accuracy. Normalization is the cheapest strategy and helps when dialect data is scarcer still, but it cannot recover entities whose spelling is unchanged.

6. Conclusion

If you have a few thousand sentences, fine-tune. If you have less, normalize. Multilingual training is a reasonable middle ground when you want one model for several dialects, but it is rarely the best choice for any single one.

Evaluating Machine Translation of Idiomatic Expressions

Sofia Andersson, Nordic Language Technology Center
Mei-Ling Chen, Harbor Institute of Technology

Abstract. Idiomatic expressions are a well-known weakness of machine translation, but they are rarely measured directly. We build a test set of 1,200 idioms in 6 language pairs, each with a literal reference and a natural reference, and we show that standard metrics reward literal translations over natural ones. We propose a simple idiom-aware metric and release the test set.

1. Introduction

Ask a machine translation system to translate "it's raining cats and dogs" and you will usually get a literal rendering that is technically correct and entirely useless. This is a well-known failure mode, but it is rarely measured directly, because standard test sets contain few idioms and standard metrics do not distinguish them.

We build a test set that does, and we show that the metrics we rely on most reward the wrong behavior.

2. Related work

Work on idiom translation has mostly been qualitative (Fadaee et al., 2018) or has focused on a single language pair. Multilingual benchmarks such as FLORES (Goyal et al., 2022) include idioms by accident, not by design.

3. Test set

We collect 1,200 idioms in 6 language pairs. For each idiom we write a literal reference, which translates the words, and a natural reference, which translates the meaning. Both references are written by native speakers.

4. Metrics

We score 8 systems with BLEU, chrF, and a learned metric. On the literal references, the ranking is sensible. On the natural references, it is not: systems that produce literal translations score higher than systems that produce natural ones.

We propose a simple idiom-aware metric that scores a translation against both references and takes the better of the two, and we release the test set and the scoring script.

Corpus Construction for Historical Orthography

Jonathan Pierce, Meridian Technical College
Helena Marquez, University of Northbridge

Abstract. We describe the construction of a corpus of historical orthography, covering 3 centuries of printed text in 2 languages. We discuss the practical problems of digitization, segmentation, and normalization, and we release the corpus and the tools we built to make it.

1. Introduction

Historical orthography is a problem for almost every digital humanities project that touches pre-standardization text. Spelling varies by author, by printer, and by year, and a search for a modern word will miss most of the relevant documents. Normalization can help, but it can also erase the very variation a researcher wants to study.

We describe the construction of a corpus that tries to preserve both.

2. Related work

Bollmann (2019) surveys rule-based and neural approaches to normalization. Piotrowski (2012) discusses the broader problem of natural language processing for historical text. Our work is mostly engineering: we build on existing tools and document the decisions we made.

3. Digitization

Our source material is 3 centuries of printed text in 2 languages, scanned from microfilm. We describe the segmentation and character-recognition steps, and the manual correction we applied to the worst pages.

4. Normalization

We provide two versions of every text: a diplomatic version that preserves the original spelling, and a normalized version that maps it to a modern standard. The normalization rules are released with the corpus, so that researchers can audit them or replace them.

5. Release

The corpus and the tools are released under permissive licenses. We hope that the decisions we made will be useful to the next project, even if the corpus itself is not.

Book Review: Statistical Methods for Text Analysis

Dr. Aisha Rahman, Coastal Research Laboratory

Statistical Methods for Text Analysis, by Lena Voss and Marcus Hale, Second Edition, 412 pages, Northbridge Press, 2025.

The first edition of this book was the standard introduction to quantitative text analysis for a decade. The second edition is a substantial revision rather than a reprint, and it is worth reading even if you own the first.

What changed

The largest change is the addition of a chapter on topic models that treats them as one tool among several rather than as the default. The authors now spend more time on when not to use a topic model, which is the right emphasis.

The chapter on classification has been rewritten around cross-validation and error analysis, and it includes a worked example that walks through a misclassified document by hand. This is the kind of exercise that separates a practitioner from a user of software.

What is missing

The book still has almost nothing to say about neural methods. For a second edition published in 2025, this is a real gap. A short chapter, even a cautious one, would have been welcome.

The exercises are also thin. Most chapters end with two or three questions that can be answered by rereading the text. More problems that require writing code would make the book more useful in a classroom.

Verdict

Recommended for graduate students and working analysts who want a careful introduction to the classical methods. Skip it if you already know them.

Call for Papers

We are accepting submissions for our first issue. Research articles, short reports, tutorials, and reviews are all welcome. See the author guidelines for formatting and submission instructions.

Deadline: 30 June 2026.