Data-augmentation pipeline for meta-templates 3-cfu Project Work

Gianluca Di Nenno, Gabriele Sinigaglia,
Master's Degree in Artificial Intelligence, University of Bologna
{ gianluca.dinenno, gabriele.sinigaglia}@studio.unibo.it

Abstract

This project focuses on enriching a dataset of meta-templates by translating them into Italian and augmenting them through BabelNet-based synonym substitution. This work constitutes a key step toward creating Bonita, an open-source Italian adaptation of the Bonito framework, thus laying the groundwork for scalable synthetic dataset generation for Italian instruction-following models. The main objective is to generate semantically diverse and linguistically accurate Italian templates that can serve as foundational resources for instruction-tuned language models. The approach combines translation, morphological and syntactic analysis, BabelNet's part-of-speech, feature-constrained synonym retrieval, and sentence embedding-based filtering to ensure contextual appropriateness. To validate the quality of the augmented data, we designed and implemented a scalable human evaluation framework based on a Human Computation paradigm. This system, deployed as an interactive CAPTCHA, collects granular user feedback on sentence correctness. The proposed analysis methodology uses control samples (Good/Bad Samples) to calibrate a reliability model that mitigates low-effort user interactions.

1 Introduction

Natural language processing systems demand robust training data to perform well across diverse domains and languages. Many tasks, however, are limited by scarce annotated data, especially in languages other than English. Expanding available resources for Italian NLP is essential to advance language technologies. Existing instruction-tuning frameworks like Bonito and widespread augmentation libraries predominantly operate in English, leaving significant gaps in multilingual dataset creation and utilization.

Textual data augmentation aims to artificially create new training examples through transformations such as synonym replacement, paraphrasing, or other more sophisticated strategies. Standard approaches like random swaps, paraphrase generation, back-translation or embedding-space transformations, improve performance, generalization, and robustness, but struggle to maintain original semantic and grammatical validity. Recent approaches focus on combining rule-based and generative techniques, filtering for label-preserving augmentations, and leveraging pre-trained language models. While many methods are effective in English, their extension to less resourced languages remains challenging due to lexicon sparsity, inaccuracies in machine translation, and limited pre trained models.(Bayer et al., 2022)

This project aims to design a pipeline for generating and evaluating Italian meta-templates suitable for instruction tuning and scalable synthetic data generation. First, existing meta-templates were translated into Italian, balancing manual and automatic methods with respect to fluency and semantic accuracy. Next, the dataset was further augmented by extracting BabelNet-constrained synonyms consistent with part-of-speech and morphological features, then filtered through multilingual embedding-based similarity checks. The method leverages the taxonomy and best practices in (Bayer et al., 2022) and integrates lessons from recent augmentation libraries such as BabelNet. The long term goal is to enable the creation of Bonita, an Italian adaptation of the Bonito framework.

Experiments were performed on several translated meta-templates taken from the ctga-v1 dataset of meta-templates. Multiple versions were produced: with and without similarity control, evaluated for resemblance and naturalness. Downstream effects on error patterns in synonym replacements, and preservation of semantic intent led us to filter the set of BabelNet synonyms with an embedding based selection for semantic plausibility and to put morphological constraints to improve the quality and usefulness of augmented examples.

2 Background

Bonito is an open-source framework designed to generate synthetic instruction-tuning datasets from unannotated text, enabling large language models to adapt to specialized domains with minimal manual annotation. Rather than relying on exhaustive human labeling, Bonito utilizes conditional task generation, remixing dataset structures and meta-templates to create diverse, task-specific instruction pairs for training. This allows researchers to synthesize lots of training examples across a wide array of NLP tasks—such as question answering, natural language inference, sentiment analysis, and summarization, showing significant improvements in zero-shot performance.(Nayak et al., 2024)

Data augmentation in NLP aims to expand training datasets and improve model generalization by creating realistic variants of existing examples. Common techniques span word-level (synonym replacement, random insertion/deletion), sentence-level (paraphrasing, sentence shuffling), and document-level transformations. Synonym replacement uses lexical resources to substitute words with their context-preserving synonyms (WordNet, BabelNet). Other approaches include doing a back-translation, where a sentence is translated into a target language and then translating it back to the original language. The resulting sentence often has a different surface form, but maintains the original meaning, providing a high quality of the word. Using pre-trained models to create new, contextually rich sentences or instruction pairs has also become a powerful approach. In fact, these models can be prompted to create new, contextually rich sentences from a given input. Another common practice is noise injection, which involves techniques like random word insertion, deletion, or swapping within a sentence. While simple to implement, these methods can sometimes break the grammatical structure or alter the meaning of the sentence, requiring careful application.

Evaluation of augmentation relies on both automated and manual strategies. Automated evaluation includes metrics that compute sematic similarities, or more specifically that computes the rate of acceptance of a new word or the rate of bad substitutions compared to good ones. In the context of this project, we primarily relied on automated metrics to guide the iterative development of our augmentation pipeline.

Despite the utility of automated metrics, they cannot fully capture the nuances of language, such as naturalness, coherence, and subtle shifts in meaning. Therefore, human evaluation is often necessary as a final quality check. This qualitative feedback is invaluable for identifying over-aggressive or semantically invalid modifications that automated metrics might miss, and for filtering the final augmented dataset to ensure the highest quality.

3 System description

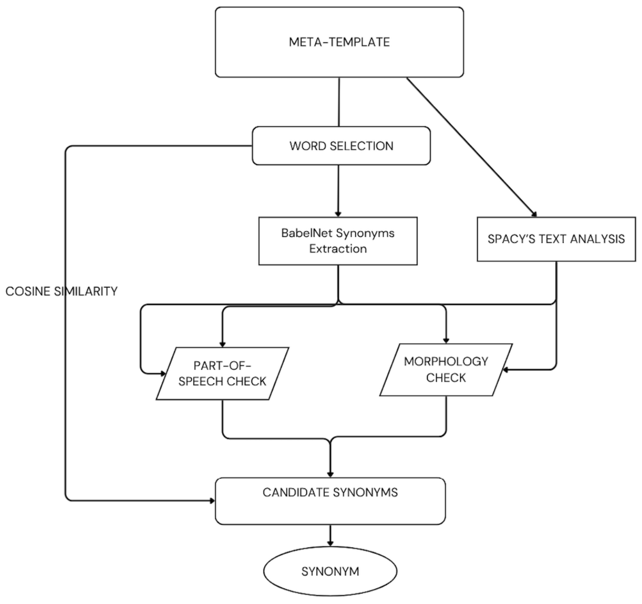

Figure 1: Enter Caption

The system developed for this project is a modular data augmentation pipeline designed to enrich an Italian dataset of meta-templates by leveraging lexical resources and modern NLP tools. At its core, the architecture is custom-built for this work and centers around a class named BabelNetAugmenter, which orchestrates the augmentation procedure.

The pipeline begins with loading and parsing the original text dataset and for each element of the meta-template the system performs linguistic analysis using spaCy's Italian language model to extract part-of-speech (POS), lemmatization, and morphological features. This is done to filter later the synonyms with the same part-of-speech and morphological features as the original words. A replacement rate is set to not change every word of the sentence, but to change at a certain rate. The system identifies candidate words by considering only content words like nouns, verbs, adjectives and adverbs and skipping articles, conjunctions, auxiliary verbs and prepositions. Words present

in a manually curated "skip list" (common verbs like essere, avere) are also excluded. Additionally, very short words (less than five characters), proper nouns, stopwords, and words starting with an uppercase letter in non-initial positions are ignored, to preserve sentence structure and meaning.
For each suitable word, the system queries BabelNet via its Python API, retrieving contextually-appropriate synonyms filtered by POS and morphological compatibility. To maximize naturalness, the algorithm uses a multilingual sentence embedding model to embed both the original context (either the whole sentence or a local window) and each candidate synonym inserted into that context. Cosine similarity is then computed between the embedding of the original sentence and each candidate-modified sentence. Only synonyms achieving a minimum threshold of similarity (e.g. above 0.5) are considered plausible replacements. The synonym achieving the highest cosine similarity to the original context is selected for substitution. If no candidate passes the similarity threshold, the original word is retained, ensuring augmentations that are semantically and contextually appropriate.
To effectively perform augmentation on Italian language templates, the system integrates three key technologies, each solving a specific subproblem critical for good quality data enrichment. First, the spaCy library and its Italian language model are used to obtain precise linguistic annotations of every word in a template. This tool provides detailed information on part-of-speech, lemma, morphological traits such as gender, number, and verb tense. This kind of linguistic analysis is crucial in Italian due to the rich morphology of the language and is used to identify both candidate words suitable for synonym substitution and to enforce agreement constraints during augmentation. Second, BabelNet acts as the core lexical resource for retrieving synonym candidates. Its rich multilingual database allows access to a broad array of Italian synonyms, filtered by the required grammatical category. BabelNet's integration ensures that the system can propose substitutions that are not only correct in meaning but also fully compatible in terms of linguistic features. Lastly, the selection of the optimal synonym for each candidate relies on sentence embedding libraries, specifically SentenceTransformers. For each substitution, the model rapidly computes multilingual sentence embeddings and uses cosine similarity to efficiently determine which synonym best preserves the original sentence's context and semantics. The embedding library was chosen for its ability to combine speed with robust contextual understanding, enabling fast, large-scale augmentation without sacrificing fluency or meaning.

3.1 A Human Computation Framework for Validation

While the augmentation pipeline is effective at generating a large volume of syntactically varied meta-templates, a robust validation stage is crucial. Automated metrics are often insufficient to guarantee the semantic coherence and natural fluency required for high-quality instruction-tuning datasets. To address this, we designed and implemented a validation framework based on the **Human Computation** paradigm, deployed as an interactive CAPTCHA. The system serves as a methodological proof-of-concept for a scalable validation pipeline, sticking to the core principles of an effective CAPTCHA:

- **Low Cognitive Load and Speed:** The primary task—validating or correcting a short, comprehensible sentence—is designed to be completed in seconds, minimizing user friction and respecting the primary goal of a CAPTCHA, which is to grant access quickly.
- **Usefulness ("CAPTCHA with a purpose"):** Every interaction in our system generates valuable, structured linguistic data and validates the augmentation work effectively turning a security measure into a distributed data annotation tool .
- **Scalability and Accessibility:** By distributing the validation effort across a potentially large number of users, the system can process datasets far more efficiently than a small team of expert annotators. The task itself does not require specialized linguistic knowledge, making it accessible to a general audience.

3.1.1 Architectural Design

The validation system is implemented as a modern web application with a decoupled client-server architecture to ensure maintainability and scalability.

- **Frontend:** An interactive **Blazor Server** application presents the validation task to the user. This choice allows for a rich, responsive user interface developed entirely in C#.

- **Backend:** An **ASP.NET Core Web API** orchestrates the system's logic. It exposes endpoints to serve preprocessed validation tasks from the database and to receive and persist user-submitted feedback.

- **Data Persistence:** A relational **SQLite** database, managed via Entity Framework Core, acts as the data store. It is designed to house 'OriginalSentences' (the full meta-templates), 'SuggestedCorrections' (aggregated user corrections), 'SubmissionLogs' (a granular record of every interaction, including metadata like interaction time), and the crucial 'CaptchaSentences' table for preprocessed tasks.

3.1.2 Data Preprocessing and Task Presentation

A significant innovation of our framework is the **offline preprocessing pipeline** designed to optimize the validation task for both user experience and data quality. Instead of presenting raw, potentially lengthy meta-templates, an automated script ('DataPreprocessor.cs') prepares the data in advance:

1. **Semantic Segmentation:** Long 'context' fields from the original meta-templates are segmented into smaller, semantically coherent sentences, primarily using punctuation (e.g., periods) as natural delimiters.

2. **Length Normalization:** Sentences that still exceed a predefined word count are further chunked into manageable lengths, ensuring the user is never overwhelmed.

3. **Index Mapping:** Crucially, each generated sub-sentence is stored in a dedicated 'CaptchaSentences' table. This record maintains a 'WordOffset', an integer that tracks the starting position of the sub-sentence's words relative to the original, complete meta-template. This ensures that any user correction can be accurately re-contextualized.

This preprocessing ensures that the task presented to the user is always concise and comprehensible, maximizing the quality of the collected feedback. A detailed discussion of the methodology used to analyze this feedback is presented in Section 6.

4 Data

The initial data for this project was sourced from the meta-templates provided by the Bonito project repository, specifically drawing on the ctga-v1 dataset. These resources include a rich set of task-oriented meta-templates organized by task type, enabling multilingual synthetic dataset generation for models trained on varied instruction formats.
For this study, dataset collection and sampling was performed in order to cover a wide array of task types. From the 16 supported NLP task types (such as question answering, sentiment analysis, paraphrase generation, textual entailment, etc.) we selected 4 unique meta-templates for each task type. These templates were initially collected and organized in a structured JSON format, with explicit labeling of task type and prompt text.
A crucial preprocessing step involved manual translation: each selected meta-template was carefully translated into Italian to ensure linguistic fidelity and naturalness. The translated templates were performed into a new JSON file, which then served as the input for the data augmentation pipeline. The translation process involved no noise injection during data creation but only a mix between careful manual translation and automated translation always manually edited to guarantee high linguistic quality and fidelity to the original meaning. This ensured the dataset was a reliable base for high-quality augmentation and evaluation. Links and further dataset information are detailed in Section 8

5 Experimental setup and results

The augmentation experiments were carried out in iterative steps, aiming to maximize grammaticality and semantic consistency in the generated Italian meta-templates.
The initial experiments used WordNet as the synonym source for Italian words. However, WordNet's coverage and flexibility are limited for Italian, often returning few synonyms and regularly failing to capture contextually relevant alternatives. Many generated sentences were awkward or included synonyms that did not fit the original meaning, motivating a shift to a richer resource.
BabelNet was adopted for its comprehensive multilingual synset graph, which extends and improves on WordNet by integrating Wikipedia and other lexical resources, using automatic map-

ping and translation to build multilingual synsets, which greatly expands coverage beyond traditional wordnets. This provided a wider and more relevant selection of Italian synonyms, better supporting complex syntactic structures and diverse word senses. Initial BabelNet-based replacements did not apply semantic or grammatical filters, so any candidate synonym could be inserted. This resulted in severe errors like verbs replaced with nouns, gender mismatches, and nonsensical outputs. These outputs highlighted the need for context and morphosyntactic constraints.

5.1 Contextual coherence

To improve contextual coherence, a semantic similarity filter was introduced to select among BabelNet candidates by embedding a local context for the target word and each synonym, then choosing the synonym with the highest cosine similarity above a minimum threshold. Emmbeddings were done using a multilingual sentence embedding model from SentenceTransformer.
The context used as target is constructed using a sliding window around the target token and this was done to embed a more semantically relevant aspect of the sentence rather than just a word. Then when the script has to pick the best synonym, it embeds the context and the list of candidate synonyms, it computes a similarity vector, takes the index of the maximum similarity, and retrieves the corresponding synonym and score. If the best similarity score is not at least equal to a minimum threshold, it returns the original word, acting as a safety valve against low-confidence substitutions. This step filtered out semantically implausible substitutions but without additional constraints it did not fix grammatical inconsistencies, and, for example, a noun with high topical relatedness can still outrank a verb, producing category mismatches and agreement errors.

5.2 POS and morphology substitution

To resolve grammatical errors, POS tags and morphological features were rigorously enforced thanks to the spaCy's Italian model, which provided reliable part-of-speech and detailed morphological information (case, gender, number, tense) for each word in the original sentence. Only BabelNet synonyms with matching POS and compatible morphological traits were considered as valid candidates before the semantic selection. This drastically reduced categorical mistakes and agreement errors, yielding sentences where substitutions respected Italian grammar and sentence structure.

6 Discussion

The augmentation experiments proved that combining good lexical resources, morphosyntactic filters and contextual similarity returns high-quality Italian meta-templates suitable for instruction tuning. However, a deeper analysis reveals the limitations of automated metrics and highlights the need for a scalable human-in-the-loop validation process. This section analyzes the quantitative results, examines error patterns through concrete examples, discusses limitations, and proposes a methodological framework for scalable Human Validation

6.1 Key Findings

The iterative refinement from WordNet to BabelNet, and from unfiltered substitution to the introduction of filters about part-of-speech, morphology and cosine-similarity progressively reduced grammatical and semantic errors. BabelNet provided broader Italian coverage than WordNet, but unfiltered synonym selection produced severe category mismatches and agreement violations. Only the addition of POS tagging and morphological feature matching eliminated most categorical and agreement mistakes, achieving the best balance between fluency and diversity.

6.2 Error Analysis and Limitations of Automated Metrics

6.2.1 Cosine Similarity only

Semantic proximity is optimized, but grammatical constraints are unchecked, resulting in a mix of both error types.

- Example 1: compagno $\rightarrow$ compagna (masculine $\rightarrow$ feminine, gender swap)
- Example 2: infanzia $\rightarrow$ infantile (noun $\rightarrow$ adjective, POS swap)

These demonstrate that a selection based only on embeddings cannot ensure linguistic validity in a rich language and for this reason grammatical filters are critical.

6.2.2 POS filtering + Cosine Similarity

This setup enforces part-of-speech, but ignores gender, number, and tense. The results are semantically reasonable but grammatically inconsistent.

Table 1: Comparison of augmentation configurations across key metrics

Configuration	**Acceptance Rate (%)**	**POS-Swap Errors (%)**	**Agreement Errors (%)**	**Correctness Rate(%)**
WordNet baseline	10,2	30,9	45,45	23,64
BabelNet + Cosine	5,7	50,8	59	10
BabelNet + POS + Cosine	6,5	28	57,1	28,5
BabelNet + Morph + Cosine	11,6	24	0	76
BabelNet + POS + Morph + Cosine	**13,4**	**15**	**0**	**85**

- Example 1: studenti → studente (plural noun → singular noun)
- Example 2: bambini → bambina (masculine plural → feminine singular)

These errors arise because POS.NOUN matches both forms, yet Italian requires number and gender agreement.

6.2.3 Morphology filtering + Cosine Similarity (no POS check)

Here, gender, number and tense are matched, but category constraints are absent, leading to POS category errors.

- Example 1: musicale → musica (adjective → noun, both feminine singular but incompatible syntactically)

6.2.4 Full system (POS + Morphology + Cosine)

The combined approach returns the best results, but rare errors persist due to lexical resource limitations.

- Example 1: nipote → cognata (both feminine singular nouns, but the substitution semantically questionable)
- Example 2: Lang → Lăng (proper-name artifact from multilingual synsets)

These outliers suggest the need for additional safeguards. However, for the purposes of data augmentation, these errors are less critical as long as the augmented sentences remain semantically and syntactically valid.

6.3 Quantitative results

Table 1 presents a quantitative comparison of different augmentation configurations evaluated on a subset of the Italian meta-template dataset. Due to BabelNet API usage limitations, the evaluation was conducted on a representative sample rather than the complete dataset of 64 templates.

The Acceptance Rate indicates the percentage of candidate synonyms that passed the minimum cosine similarity threshold, The WordNet baseline shows a low acceptance rate of 10.2 percent, reflecting its limited coverage for Italian synonyms. Introducing BabelNet with cosine similarity filtering alone yields only 5.7 percent, suggesting that semantic similarity without grammatical constraints is overly restrictive. However, configurations incorporating POS filtering show notably higher acceptance rates, with the combined BabelNet + POS + Morph + Cosine achieving 13.4 percent, demonstrating that grammatical filtering enables more confident synonym selection while maintaining quality.

POS-Swap Errors measure instances where a word from one grammatical category is incorrectly replaced with a word from another category. The baseline and similarity-only configurations exhibit high error rates (30.9% and 50.8%, respectively), confirming that semantic proximity alone cannot prevent categorical mismatches.

The introduction of POS filtering dramatically reduces these errors to 28% and eventually to 15% in the full configuration, though residual errors persist. These remaining POS violations can be attributed to several factors: imperfect POS tagging by spaCy on morphologically complex or rare Italian forms or inconsistencies in BabelNet's POS annotations since we are comparing BabelNet POS with Spacy's POS.

Agreement Errors capture violations of Italian morphological agreement like gender, number, person, or tense mismatches that render substitutions grammatically incorrect. Configurations without morphological filtering show substantial error rates (45.45% for baseline, 59% for BabelNet + Cosine), highlighting the critical importance of morphology in Italian. The addition of morphology-only

checks reduces agreement errors to 0% in that specific configuration, and the full system maintains 0% agreement errors as well, demonstrating that explicit morphological feature matching successfully enforces Italian grammatical constraints when implemented correctly.
The Correctness Rate represents the overall percentage of augmented sentences that are both semantically plausible and grammatically valid according to manual inspection. The full BabelNet + POS + Morph + Cosine configuration achieves the highest correctness at 85%, a substantial improvement over simpler approaches. This indicates that while the combined filtering strategy significantly enhances quality, approximately 15% of outputs still contain minor issues-such as rare synonym choices, slight semantic drift in context, or the before mentioned residual POS errors-underscoring the inherent challenges of automated augmentation in morphologically rich languages and the potential value of human-in-the-loop validation for production use.

6.3.1 Limitations

The error analysis reveals that these metrics are most of the time insufficient to detect issues of pragmatic appropriateness and natural fluency. Some limitations to this approach might be the fact that some synsets contain rare, archaic, or cross-lingual artifacts, and sense boundaries are not always precise, leading to occasional mismatches. Also, spaCy's Italian models occasionally misanalyze rare or morphologically irregular forms, which can propagate errors into synonym filtering.
This confirms that human evaluation is not just a final check, but a necessary component for building high-quality instruction-tuning datasets. The primary limitation of the augmentation pipeline is therefore not in its ability to generate variations, but in its capacity to self-evaluate them reliably. This leads to the central challenge: **How can we implement human evaluation at a scale that matches the data generation process?**

6.4 A Methodological Framework for Scalable Human Validation

The error analysis confirms that automated metrics are insufficient, highlighting the central challenge: implementing human evaluation at scale. To solve this, we propose and have implemented a complete methodological framework, embodied by our Human Computation system, designed not just to collect data, but to prepare it intelligently and analyze it in a statistically robust manner.

6.4.1 System Architecture and Data Preprocessing

The framework's architecture is engineered for scalability and data quality. A backend **ASP.NET Core API** serves tasks to an interactive **Blazor Server** frontend, with all data persisted in a **SQLite** database.

A key innovation is the **offline preprocessing pipeline**. Instead of showing raw meta-templates, a script segments long texts into coherent sentences and normalizes their length. Crucially, each generated sub-sentence is stored in a dedicated 'CaptchaSentences' table, maintaining a 'WordOffset' that maps its words back to the original meta-template. The frontend further enhances data quality with a **regex-based tokenizer** that separates clickable words from non-interactive punctuation, ensuring user feedback is precise.

6.4.2 Data Collection and Stratification

To enable a rigorous analysis and calibrate a reliability model, the dataset is stratified. Each entry in the 'OriginalSentences' table is partitioned into one of three categories using a 'ControlStatus' flag:

- **Control Group (Good Samples):** A manually verified subset of sentences guaranteed to be grammatically and semantically correct.
- **Control Group (Bad Samples):** A subset of sentences with deliberately introduced, unambiguous errors (e.g., typos, incorrect words).
- **Experimental Group (Unverified Samples):** The remaining majority of augmented meta-templates whose quality is unknown.

Every user submission is then recorded as a new entry in the 'SubmissionLogs' table, capturing a rich set of metrics essential for our reliability model:

1. **User Action:** A boolean flag ('WasMarkedAsCorrect') distinguishing a positive validation from a correction.
2. **Correction Data:** For corrections, the system stores the word's 'WordPosition' (recalculated using the 'WordOffset'), the 'OriginalWord', and the user's 'SuggestedWord'.
3. **Interaction Time:** The time in seconds elapsed between task display and

feedback submission (‘InteractionTimeInSeconds‘), used as a proxy for cognitive effort.

6.4.3 Model Calibration and Reliability-Weighted Evaluation

This detailed, stratified data allows for the calibration of a robust reliability model. An automated analysis function (‘AnalyzeStatisticsAsync‘) uses the control groups to establish baseline user behavior, including the "lazy clicking" rate (from ‘Bad Samples‘) and the benchmark for genuine approval (from ‘Good Samples‘).

With these calibrated thresholds, the framework can evaluate unverified sentences and corrections. The reliability of a user-suggested correction is calculated using a weighted score that considers its popularity (‘SubmissionCount‘), the context of the source sentence, and the user’s effort (interaction time).

6.4.4 Automated Feedback Loop and Context Reconstruction

The framework culminates in an automated feedback loop. Corrections achieving a high score are flagged for promotion. The framework’s most powerful feature is its ability to **reconstruct the full meta-template** thanks to the ‘WordOffset‘, accurately applying corrections made on sub-sentences back to the original text. This creates a continuous feedback loop where the dataset’s quality improves over time through collective human intelligence.

7 Conclusion

In this project, we have iteratively refined a data augmentation pipeline for Italian meta-templates, a crucial step toward creating Bonita, an Italian adaptation of the Bonito framework. Our main finding is that for a morphologically rich language like Italian, a combination of lexical resources (BabelNet), semantic filtering (cosine similarity), and grammatical constraints (POS and morphology matching) is essential to produce high-quality augmentations. While initial experiments with simpler methods like WordNet or unfiltered synonym replacement produced grammatically incorrect or semantically inconsistent sentences, our final system achieved an 85% correctness rate on a manually evaluated subset. The most surprising observation was the dramatic impact of morphological filtering, which single-handedly eliminated all agreement errors—a testament to its necessity in this linguistic context. The primary limitation of our work is the difficulty of automatically verifying the quality of generated data at scale; automated metrics, while useful for guidance, cannot fully capture subtle shades of the language involved. This led us to design an implementation of a scalable, human-in-the-loop validation framework deployed as an interactive CAPTCHA. By distributing the validation task across many users, this system transforms a routine security check into a powerful data annotation tool. Its architecture, featuring offline sentence segmentation and a robust feedback mechanism with control samples, is engineered to collect reliable, fine-grained corrections while maintaining a low cognitive load for users.
A promising direction for future work is to fully deploy this CAPTCHA system to validate the entire augmented dataset, creating a continuous feedback loop where collective human intelligence iteratively refines the data.

8 Links to external resources

Insert here:

- GitHub repo with our code; https://github.com/sini-g/NLP-metatemplate-AUG.git
- link to the dataset; https://github.com/BatsResearch/bonito; https://huggingface.co/datasets/BatsResearch/ctga-v1;

References

Markus Bayer, Marc-André Kaufhold, and Christian Reuter. 2022. A survey on data augmentation for text classification. *ACM Computing Surveys*, 55(7):1–39.

Nihal V. Nayak, Yiyang Nan, Avi Trost, and Stephen H. Bach. 2024. Learning to generate instruction tuning datasets for zero-shot task adaptation.