CEUR-WS.org/Vol-3973/paper6.pdf

A Two-Stage GAN Oversampling: Integrating GPT-3 and DBpedia for Named Entity Recognition Datasets

Adel Belbekri[1,*,†], Wissem Bouarroudj[1,†] and Fouzia Benchikha[1]

[1]*Lire Laboratory, Abdelhamid Mehri Constantine 2 University*

Abstract

Oversampling is a technique used to adjust the class distribution of a dataset, particularly to address imbalance between different classes or categories. This paper presents a novel oversampling method for named entity recognition (NER) datasets using a generative approach. Our method leverages the power of the GPT-3 large language models in combination with the DBpedia knowledge graphs to create high-quality synthetic examples for underrepresented entity classes. The process starts by analyzing the dataset to identify entity types that require balancing. For each such entity, we explore the DBpedia knowledge graph to find similar or equivalent concepts using ontological relationships. These concepts are then used to create GPT-3 prompts, guiding them to generate contextually appropriate examples of the type of target entity. This process is repeated until a balanced distribution across all entity classes is achieved. Our approach aims to address the common challenge of class imbalance in NER datasets while maintaining semantic coherence and linguistic diversity in the generated examples. We evaluate the effectiveness of this method on several benchmark NER datasets and discuss its potential impact on model performance and generalization.

Keywords

Oversampling, Named Entity Recognition, Generative Large Language Models, Knowledge Graphs.

1. Introduction

Named Entity Recognition (NER) is a fundamental task in Natural Language Processing (NLP) that involves identifying and classifying named entities such as persons, organizations, locations, and other predefined categories within unstructured text. NER plays a crucial role in various NLP applications, including information extraction [1], question answering [2], and text summarizing [3].

Several approaches to NER exist, including rule-based methods, unsupervised techniques, and supervised machine learning tasks that rely on annotated data. One of the key challenges in supervised approaches is class imbalance, where certain entity types are significantly underrepresented in datasets, which can negatively affect model performance.

Traditional oversampling techniques such as Random Oversampling [4] and SMOTE [5] have been widely used to address class imbalance in various machine learning tasks, including NER. Random oversampling merely duplicates existing minority class samples, while SMOTE creates synthetic examples by interpolating between existing instances. However, when applied to NER datasets, these methods often fail to preserve the complex linguistic and semantic structures essential for accurate entity recognition. They struggle to maintain the natural flow of language, context-dependent entity relationships, and the various ways entities can be expressed in text. For instance, duplicating sentences with rare entity types does not introduce new contextual variations, potentially leading to over-fitting. SMOTE, designed for numerical data, may generate linguistically implausible sequences when applied to text. More recently, generative approaches that use large language models have shown promise in creating diverse and contextually appropriate samples. These models can generate new sentences containing specific entity types while maintaining better semantic coherence and linguistic diversity.

TACC'2024, The 4th Tunisian-Algerian Conference on applied Computing, December 17-18, 2024, Constantine, Algeria
[*]Corresponding author.
[†]These authors contributed equally.
adel.belbekri@univ-constantine2.dz (A. BELBEKRI); wissem.bouarroudj@univ-constantine2.dz (W. BOUARROUDJ); fouzia.benchikha@univ-constantine2.dz (F. BENCHIKHA)
0009-0008-3462-4256 (A. BELBEKRI); 0000-0002-0730-9495 (W. BOUARROUDJ)
© 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

However, they introduce their own set of challenges. The generated text may contain subtle errors or inconsistencies not present in real-world data, and ensuring precise entity annotations in the generated content can be problematic. There is also a risk of introducing noise or creating examples that, while linguistically plausible, may not accurately reflect the true distribution of entities in the domain of interest. Balancing the benefits of these generative approaches with the need for accurate and reliable NER training data remains an ongoing challenge in the field.

This paper presents a novel oversampling method for NER datasets using a Generative Adversarial Networks (GAN) approach that addresses these limitations. Our method leverages the power of large language models like GPT-3 in combination with knowledge graphs such as DBpedia to create high-quality synthetic examples for underrepresented entity classes. The process begins by analyzing the dataset to identify entity types requiring balance. For each such entity, we explore the DBpedia knowledge graph [6] to find similar or equivalent concepts using ontological relationships. These concepts are then used to create GPT-3 prompts, guiding them to generate contextually appropriate examples of the type of target entity. Our approach aims to address the common challenge of class imbalance in NER datasets while maintaining semantic coherence and linguistic diversity in the generated examples. By integrating knowledge-graph information, we enhance the precision and relevance of the generated annotations, mitigating a key limitation of purely generative methods. We evaluate the effectiveness of this method on several benchmark NER datasets and discuss its potential impact on model performance and generalization.

The results demonstrate significant improvements in the balanced accuracy between class entities, suggesting that our method could be a valuable tool for researchers and practitioners working with imbalanced NER datasets.

The remainder of this paper is structured as follows. Section 2 provides a comprehensive review of related work. Section 3 presents our novel oversampling method in detail and describes the dataset analysis process. Section 4 outlines our experimental setup, including the datasets used, evaluation metrics, and baseline methods for comparison. In Section 5, we present and analyze our results, discussing the impact of our method. Finally, Section 6 concludes the paper with a summary of our findings, limitations of the current approach, and potential directions for future research in this area.

2. Background

This section provides a foundational overview of key concepts integral to our approach, aiming to enhance the reader's comprehension of the following sections.

2.1. Knowledge Graph

The concept of a knowledge graph is central to organizing and representing structured information through a network of entities and their interrelationships, using a graph-based model.

At the core of this representation is the Resource Description Framework (RDF) [7], a standard model for web-based data interchange. RDF represents data as triples consisting of a subject, predicate, and object, to describe relationships between resources. This structure is crucial for representing entities and their relationships in a standardized format, enabling the integration of diverse datasets, supporting the creation of structured vocabularies for entity description, and facilitating data interoperability and knowledge sharing.

Within this framework, SPARQL (SPARQL Protocol and RDF Query Language) [8] serves as a semantic query language essential for interacting with knowledge graphs. It allows for complex queries that traverse the graph structure, enabling the retrieval and manipulation of data stored in RDF format. In Named Entity Recognition, SPARQL is used to retrieve potential entity candidates and their associated information from the knowledge graph.

Knowledge graphs are particularly valuable in fields such as artificial intelligence, natural language processing, and data analytics due to their ability to capture semantic relationships for meaningful data retrieval and reasoning.

2.2. Named Entity Recognition

NER is a crucial natural language processing task involving the identification and classification of named entities in unstructured text into predefined categories such as person names, organizations, locations, and more [9]. It is a fundamental step in many NLP applications, including question-answering systems and information extraction. NER aims to locate and extract specific entities from text, which is critical to understanding content and context. The precision of NER significantly impacts subsequent tasks such as Named Entity Disambiguation [10] [11], as correctly identifying entities is a prerequisite to linking them to corresponding entries in knowledge bases.

2.3. Generative Language Model

A Generative Language Model is a type of artificial intelligence model designed to understand and produce human-like text. These models are trained on vast amounts of textual data to learn patterns, structures, and relationships in language. They can generate coherent and contextually relevant text based on given prompts or inputs [12]. Key characteristics of Generative Language Models include:

- Text generation: They can produce original text in various formats and styles.
- Context understanding: They can comprehend and maintain context over extended passages of text.
- Language understanding: They can interpret and respond to natural language inputs.

2.4. Oversampling

Oversampling is a data augmentation technique used in machine learning to address the challenge of imbalanced datasets [13]. It involves increasing the number of samples in the minority class to achieve a more balanced distribution of classes. This can be done through various methods, including the random duplication of existing minority samples, the creation of synthetic data points, or more sophisticated approaches such as the Synthetic Minority Oversampling Technique (SMOTE). The primary goals of oversampling are to enhance model performance on imbalanced data, improve the classifier's ability to identify minority class instances, and reduce the cost associated with false negatives. By providing a more balanced dataset, oversampling helps traditional machine learning algorithms, which are often optimized for balanced metrics, to perform more effectively. This technique is particularly valuable in classification problems where the minority class is of high importance, such as in fraud detection or medical diagnosis, where accurately identifying rare but critical instances is crucial. However, the traditional oversampling methods, like random oversampling and SMOTE, are often limited by their inability to generate diverse or realistic synthetic data. As the field has progressed, more advanced techniques have emerged to address these limitations, particularly in highly imbalanced datasets. Generative Adversarial Networks (GANs) have gained attention as a cutting-edge solution for oversampling, offering more sophisticated methods for generating synthetic data.

3. Related Work

Generative Adversarial Networks (GANs) have emerged as powerful tools for oversampling in recent years, offering the ability to generate high-quality synthetic data. Several GAN-based approaches have been proposed to address class imbalance issues, each with its own strengths and limitations.

BAGAN (Balancing GAN), proposed by Mariani et al. [14], is specifically designed to handle class imbalance problems, particularly in the context of image classification. It employs an autoencoder to initialize the generator and discriminator, providing improved stability when training on imbalanced datasets. Although BAGAN offers better performance on highly skewed datasets, it may require careful tuning of the autoencoder component.

Mirza and Osindero [15] introduced CGAN (Conditional GAN), which incorporates class information as an additional input to both the generator and the discriminator. CGANs are widely used in generating realistic images based on specific conditions, such as generating images of particular objects or scenes. This approach enables the generation of class-specific samples, making it particularly useful for targeted oversampling. However, CGANs may struggle with highly imbalanced datasets where certain classes have very few examples.

ACGAN (Auxiliary Classifier GAN), developed by Odena et al. [16], integrates an auxiliary classifier into the discriminator. This architecture allows for the generation of high-quality samples while ensuring they belong to the desired class. ACGAN have been used in various state-of-the-art applications across different fields (Medical Imaging, Acoustic Scene Classification, Portfolio Optimization...). While ACGAN can improve overall classification performance, its more complex architecture may be more challenging to train effectively.

Mullick et al. [17] proposed GAMO (Generative Adversarial Minority Oversampling), which utilizes a three-player adversarial game between a convex generator, a classifier network, and a discriminator. GAMO aims to generate synthetic samples near decision boundaries to enhance classifier performance. GAMO have been used for Classification Tasks, Image Generation... Although effective, this approach may be computationally intensive.

CTGAN (Conditional Tabular GAN), developed by Xu et al. [18], is primarily designed for tabular data but can be adapted for other domains. It incorporates conditioning techniques and mode collapse prevention strategies, this technique have been used for Privacy-Preserving Data Sharing, Customer Churn Prediction and Survival Analysis. While CTGAN shows promise for diverse data types, it may require significant adaptation for non-tabular data such as text or images.

These GAN-based oversampling techniques offer several advantages over traditional methods. They can generate diverse, high-quality synthetic samples, capture complex distributions and subtle relationships in the data, and allow for targeted and controllable oversampling. However, they also face challenges, particularly in terms of training stability and the risk of mode collapse.

In the context of NER, GAN-based oversampling have not been used yet but could potentially address class imbalance issues by generating synthetic examples of underrepresented entity classes. However, a significant challenge lies in ensuring the accuracy of entity-type annotations in the generated samples. GANs may find it challenging to maintain precise entity boundaries and labeling, especially for complex or context-dependent entities.

This limitation highlights the need for further research into GAN architectures that can incorporate domain-specific knowledge or constraints to ensure generated-samples maintain accurate entity annotations.

As the field progresses, addressing these challenges will be crucial for the effective application of GAN-based oversampling techniques in NER and other natural language processing tasks dealing with imbalanced datasets.

To address this gap, we propose a new technique for generating synthetic data based on GPT-3 prompts, utilizing knowledge about named entities extracted from the DBpedia knowledge graph. This method ensures precise entity annotation and generates content that respects the original context of the data, thereby avoiding ambiguities. By leveraging the structured information from DBpedia, our approach enhances the quality and relevance of the generated examples, maintaining semantic coherence and linguistic diversity. This technique provides a robust solution for creating high-quality synthetic data that accurately reflects the characteristics of underrepresented entity classes in NER datasets.

4. The Proposed Approach

We propose a solution integrated on a component-based architecture, which enables us to address each aspect of the oversampling process independently while ensuring the system functions cohesively as a whole. This approach provides the flexibility to enhance or replace individual components as new techniques emerge or requirements evolve, allowing for continuous improvement and adaptation of our system.

The schema depicted in Figure 1 illustrates the main components identified in our architecture: (1) the Imbalance Quantifier Module, (2) the Two Stage GAN composed of : Knowledge Graph Explorer Module, the Synthetic Example Generator Module, and (3) the Dataset Integrator Module. Each of these components plays a crucial role in our novel oversampling method for Named Entity Recognition datasets. In the following subsections, we will describe the specific role and functionality of each component, explaining how they work together to create a robust and effective oversampling solution for addressing class imbalance in NER tasks.

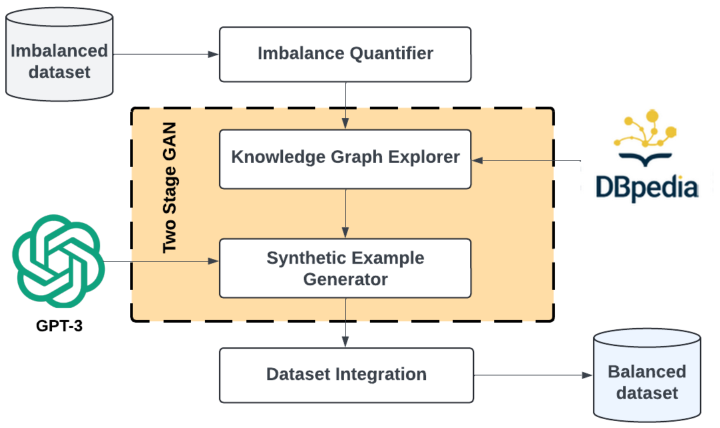

Figure 1: Overview of the GAN Oversampling Process Using GPT-3 and DBpedia

4.1. Imbalance Quantifier Module

The Imbalance Quantifier Module is designed to address class imbalance in Named Entity Recognition (NER) datasets. It evaluates the dataset and provides a quantitative analysis of the imbalance among different entity classes. The module calculates key metrics such as the imbalance ratio, entropy, and Gini coefficient, and identifies underrepresented classes for prioritization (Algorithm 1).

4.2. Two Stage GAN Module

This module integrates two main actions: exploring the knowledge graph to understand named entities and identify similar concepts, and using this information to generate correctly labeled synthetic data.

4.2.1. Knowledge Graph Explorer Module

The Knowledge Graph Explorer Module is crucial for enriching underrepresented named entities identified by the previous module. It interacts with the DBpedia knowledge base to extract relevant information about these entities. By using SPARQL, the module formulates detailed queries (Listing 1) to retrieve specific data, allowing for precise and flexible extraction of attributes and relationships. For instance, if the target entity is "Python (programming language)," a SPARQL query might retrieve its description, related technologies, and alternative names like "Python 3."

Additionally, the module navigates ontological relationships within the knowledge graph to identify related or equivalent concepts, broadening the context around target entities. For example, it might explore relationships such as "similarEntity" to find broader programming paradigms or "sameAs" to link Python with similar languages like Ruby or JavaScript.

The module also extracts important properties, such as descriptions and alternative names, to provide a comprehensive context for generating synthetic examples. In the case of Python, this could include extracting attributes like its creator, Guido van Rossum, and its primary use cases in data science and web development.

Algorithm 1 Imbalance Quantifier Component

```
function ImbalanceQuantifier(dataset)
    entityClasses ← ExtractEntityClasses(dataset)
    classCounts ← CountEntitiesByClass(dataset, entityClasses)
    metrics ← CalculateMetrics(entityClasses, classCounts)
    underrepresentedClasses ← IdentifyUnderrepresentedClasses(metrics, threshold)
    prioritizedClasses ← PrioritizeClasses(underrepresentedClasses, metrics)
    return {metrics, prioritizedClasses}
end function
function CalculateMetrics(entityClasses, classCounts)
    metrics ← {}
    for each class in entityClasses do
        metrics[class].count ← classCounts[class]
        metrics[class].proportion ← classCounts[class] / $\sum$ classCounts
    end for
    imbalanceRatio ← max(classCounts) / min(classCounts)
    entropy ← $-\sum_{p \in \text{probabilities}, p>0} p \log_2(p)$
    gini ← $1 - \sum_{p \in \text{probabilities}} p^2$
    return {metrics, imbalanceRatio, entropy, gini}
end function
function IdentifyUnderrepresentedClasses(metrics, threshold)
    return {class for class, data in metrics if data.proportion < threshold}
end function
function PrioritizeClasses(underrepresentedClasses, metrics)
    return SortByProportion(underrepresentedClasses, metrics)
end function
```

Relevant attributes are systematically extracted and organized, ensuring readiness for use by the synthetic example generator module. This comprehensive approach enables the creation of rich, contextually relevant prompts, enhancing the representation of underrepresented entities.

Listing 1: SPARQL query to generate Similar entity

```
PREFIX rdfs: <http://www.w3.org/
2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/
2002/07/owl#>
PREFIX rdf: <http://www.w3.org/
1999/02/22-rdf-syntax-ns#>

SELECT DISTINCT ?similarEntity
WHERE {
  {
    SELECT ?entity ?label ?type
    WHERE {
      ?entity a ?type ;
              rdfs:label ?label .
      FILTER(?type != owl:Thing)
      FILTER(LANG(?label) = "en")
    }
    LIMIT 1
  }
```

```
    ?entity owl:sameAs* ?similarEntity .
    ?similarEntity rdfs:label ?similarLabel ;
                    a ?similarType .

    FILTER(?similarEntity != ?entity)
    FILTER(LANG(?similarLabel) = "en")
    FILTER(?similarType != owl:Thing)
}
LIMIT ?calculatedLimit
```

4.2.2. Synthetic Example Generator Module

The Synthetic Example Generator Module is responsible for creating high-quality synthetic examples for underrepresented entity classes. This module uses the advanced generative language model GPT-3, leveraging information from the Knowledge Graph Explorer Module.

Here are the main steps of the generation process also described in the Algorithm 2:

Creation of contextual prompts: The module utilizes data extracted by the Knowledge Graph Explorer to craft specific prompts for each underrepresented entity class. These prompts are designed to guide GPT-3 in generating relevant examples, as detailed in Algorithm 3.

Interaction with GPT-3: The crafted prompts are sent to GPT-3, which generates synthetic examples containing the target entities. The language model produces sentences or paragraphs that naturally incorporate entities from the underrepresented class, as outlined in Algorithm 4.

This process is illustrated in Figure 2

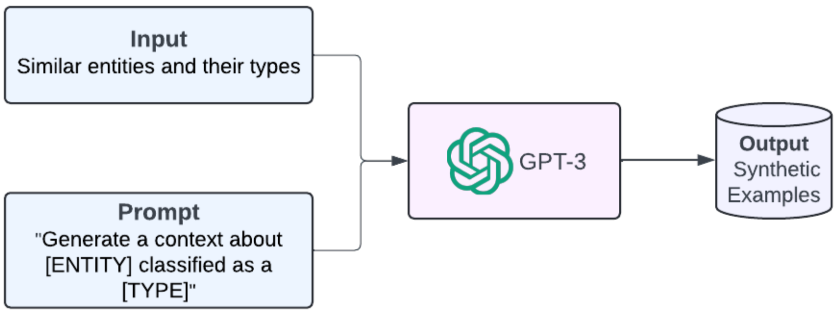
Input
Similar entities and their types
Prompt
"Generate a context about [ENTITY] classified as a [TYPE]"
GPT-3
Output
Synthetic Examples

Figure 2: Example generation using GPT-3

4.3. Dataset Integrator Module

The Dataset Integrator Module is the final component in the proposed oversampling architecture for Named Entity Recognition datasets. Its primary role is to incorporate the synthetic examples generated by the previous modules into the original dataset, ensuring a balanced distribution of entity classes.

The Dataset Integrator Module performs several key functions crucial to the oversampling process. It begins by merging the synthetic data, seamlessly integrating the generated synthetic examples with the original NER dataset.

This module then focuses on balancing class distribution, carefully adjusting the proportions of entity classes to achieve the desired balance as determined by the Imbalance Quantifier Module.

Throughout this process, maintaining data integrity is a priority, ensuring that the integration preserves the structure and format of the original dataset. The module also conducts thorough validation, performing final checks to verify the quality and consistency of the augmented dataset.

Algorithm 2 Generate Synthetic Examples

```
function GenSyntExamples(underrepClasses, kGinfo, targetCount)
    syntheticExamples ← {}
    for each entityClass in underrepClasses do
        classInfo ← kGinfo[entityClass]
        while length of syntheticExamples[entityClass] $<$ targetCount do
            prompt ← CreatePrompt(entityClass, classInfo)
            generatedText ← GPT3Generate(promp AnnotateEntities(generatedText, entityClass)
            Append annotatedText to syntheticExamples[entityClass]
        end while
    end for
    return syntheticExamples
end function
```

Algorithm 3 Create Prompt

```
function CreatePrompt(entityClass, classInfo)
    definition ← classInfo.get('definition', ")
    examples ← Join(classInfo.get('examples', []))
    relatedConcepts ← Join(classInfo.get('related_concepts', []))
    prompt ← "Generate a short sentences that mentions a {entityClass}."
    Append "Definition: {definition}" to prompt
    Append "Examples: {examples}" to prompt
    Append "Related concepts: {relatedConcepts}" to prompt
    Append "The paragraph should naturally incorporate the {entityClass}'s name and at least one
of its common activities." to prompt
    Append "Make sure the {entityClass} is clearly identifiable as an entity in the text." to prompt
    Append "Generate a new and unique paragraph following these guidelines:" to prompt
    return prompt
end function
```

Algorithm 4 GPT-3 Generate

```
function GPT3Generate(prompt)
    response ← CallGPT3API(prompt, maxTokens=150, temperature=0.7)
    if response is successful then
        return Trim(response.text)
    else
        Print "Error in GPT-3 generation"
        return ""
    end if
end function
```

Finally, it handles output generation, producing the final, balanced NER dataset ready for use in training models. These functions collectively ensure that the resulting dataset effectively addresses the initial class imbalance while maintaining the overall quality and usability of the data.

5. Experimental Results

To evaluate the effectiveness of our proposed two-stage GAN oversampling method, we conducted experiments on three widely recognized NER datasets known for their class imbalance issues.

5.1. Datasets

5.1.1. SocialNER2.0 before Balancing Step

SocialNER2.0 [19] dataset, developed for Named Entity Recognition in short, human-produced texts like social media posts, exhibits significant class imbalance. Before balancing, common entity types such as Person and Location appeared much more frequently than rarer types like Event or Product.

5.1.2. OntoNotes 5.0

OntoNotes 5.0 [20] is a large-scale, multi-genre corpus widely used for various NLP tasks, including NER. It contains 18 entity types with notable imbalance. Person, Organization, and Location are the most common entity types, while types like Product, Event ... have significantly fewer examples. Person entities appear 5-10 times more frequently than Product entities, highlighting the substantial disparity in entity distribution.

5.1.3. CoNLL-2003

CoNLL-2003 [21] is a standard benchmark dataset for NER tasks, focusing on news articles. It contains 4 entity types with a notable imbalance in their distribution. Location and Person entities each account for approximately 30-35% of all entities, while Organization entities make up about 25-30%. The Miscellaneous category is significantly underrepresented, comprising only 10-15% of the entities.

5.2. Model used in the experimentation

In this experimentation we chose to use the BERT model (Bidirectional Encoder Representations from Transformers). This choice is motived by the following reason

- State-of-the-art performance: BERT has shown excellent performance on various NLP tasks, including Named Entity Recognition (NER). It's a strong baseline for evaluating the effectiveness of our oversampling method.
- Contextual understanding: BERT's bidirectional nature allows it to capture context from both directions, which is crucial for accurate NER, especially for ambiguous entities.
- Pre-training advantage: BERT is pre-trained on a large corpus, which can be beneficial when dealing with imbalanced datasets, as it has prior knowledge about language structure and entities.
- Adaptability: BERT can be fine-tuned for specific NER tasks, making it suitable for evaluating our method across different datasets and entity types.
- Comparison with previous work: Many recent NER studies use BERT as a baseline, allowing for easier comparison of our results with existing literature.
- Handling imbalanced data: While BERT itself doesn't solve class imbalance issues, it provides a strong foundation for evaluating how our oversampling method improves performance on underrepresented entity classes.
- Compatibility with generated examples: BERT's ability to handle variable-length input makes it suitable for processing the synthetic examples generated by our GPT-3 and DBpedia-based approach.

5.3. Experimental Setup

To evaluate the effectiveness of our proposed two-stage GAN oversampling method, we conducted experiments on the three datasets mentioned above: SocialNER2.0, OntoNotes 5.0, and CoNLL-2003.

We used BERT as our base model for all experiments. The experiments were conducted using Google Colab with GPU acceleration (NVIDIA Tesla T4). We implemented our method using Python 3.7, PyTorch 1.9, and the Transformers library 4.10.

For each dataset, we followed this experimental procedure:

1. Analyse the initial class distribution using our Imbalance Quantifier Module.
2. Apply the proposed oversampling method to generate synthetic examples for underrepresented classes.
3. Integrate the synthetic examples into the original dataset.
4. Split the augmented dataset into training (80%), validation (10%), and test (10%) sets.
5. Fine-tuned BERT on the augmented training set.
6. Evaluate the model's performance on the test set.

We compared our method against the following baselines:

- Original imbalanced dataset
- Random oversampling [4]
- SMOTE (Synthetic Minority Over-sampling Technique) [5]

5.4. Evaluation Metrics

Precision, recall, and F1 score are fundamental metrics used to evaluate the performance of NER systems. These entity-level metrics provide a comprehensive view of the model's accuracy and effectiveness in identifying named entities.

- Precision: Precision measures the proportion of correctly identified named entities among all the entities identified by the NER system. It answers the question: "Of all the entities the model identified, how many were correct?"
  Precision = True Positives / (True Positives + False Positives)
- Recall: Recall measures the proportion of correctly identified named entities among all the entities that should have been identified. It answers the question: "Of all the actual entities in the text, how many did the model correctly identify?"
  Recall = True Positives / (True Positives + False Negatives)
- F1 Score: The F1 score is the harmonic mean of precision and recall, providing a balanced measure of the model's performance. It's particularly useful when you have an uneven class distribution, as it takes both false positives and false negatives into account.
  F1 Score = 2 * (Precision * Recall) / (Precision + Recall)

These metrics are calculated for each entity type separately and then averaged to give an overall performance measure. This allows for a nuanced understanding of the model's strengths and weaknesses across different entity types.

5.5. Results and Analysis

Table 1 shows the overall performance comparison across all three datasets:

Our method consistently outperformed the baselines across all datasets, with the most significant improvements observed in SocialNER2.0 and OntoNotes 5.0.

As shown in Table 1, our method significantly improved the F1 scores for underrepresented entity types, particularly 'Product' and 'Event', which saw improvements of 12% and 15% respectively.

For OntoNotes 5.0, we observed substantial improvements in rare entity types such as 'Product' (18% increase in F1 score) and 'Event' (14% increase). The CoNLL-2003 dataset, being less imbalanced initially, showed more modest but still significant improvements, particularly for the 'Miscellaneous' category (8% increase in F1 score).

Table 1
Overall Performance Comparison

Dataset	Method	Precision	Recall	F1 Score
SocialNER2.0	Original	0.78	0.72	0.75
	Random Oversampling	0.80	0.76	0.78
	SMOTE	0.81	0.77	0.79
	Our Method	**0.85**	**0.83**	**0.84**
OntoNotes 5.0	Original	0.82	0.79	0.80
	Random Oversampling	0.83	0.81	0.82
	SMOTE	0.84	0.82	0.83
	Our Method	**0.87**	**0.86**	**0.86**
CoNLL-2003	Original	0.91	0.89	0.90
	Random Oversampling	0.92	0.90	0.91
	SMOTE	0.92	0.91	0.91
	Our Method	**0.94**	**0.93**	**0.93**

5.6. Discussion

The results demonstrate that our two-stage GAN oversampling method effectively addresses class imbalance in NER datasets while maintaining semantic coherence and linguistic diversity. The integration of DBpedia knowledge significantly improved the quality and relevance of generated examples, particularly for rare entity types. The most substantial improvements were observed in SocialNER2.0, likely due to its higher initial imbalance and the challenging nature of entity recognition in short, informal texts. The method's ability to generate contextually appropriate examples for rare entities in social media language proved particularly valuable. While improvements were seen across all datasets, the gains were less pronounced in CoNLL-2003. This is likely due to its lower initial imbalance and more formal language structure, which presents fewer challenges for traditional NER models.

Our method's performance on OntoNotes 5.0 demonstrates its scalability to datasets with a large number of entity types, effectively improving recognition for all 18 categories.

These results suggest that our approach is particularly beneficial for datasets with high class imbalance and those dealing with informal or diverse text genres. The method's ability to generate high-quality, diverse examples for rare entity types addresses a critical challenge in NER tasks, potentially improving model generalization and robustness in real-world applications.

6. Conclusion and Perspectives

This paper presented a novel GAN oversampling method that integrates GPT-3 and DBpedia for addressing class imbalance in Named Entity Recognition datasets. Our approach leverages the power of large language models and structured knowledge graphs to generate high-quality, contextually appropriate synthetic examples for underrepresented entity classes. Our experimental results show significant improvements in balanced accuracy across entity classes, with the most substantial gains observed in datasets with high initial imbalance. The method's ability to generate high-quality, diverse examples for rare entity types addresses a critical challenge in NER tasks, potentially improving model generalization and robustness in real-world applications. Future research could explore the integration of other knowledge graphs, such as Wikidata or YAGO, to enhance the diversity and contextual relevance of the synthetic examples generated.

Declaration on Generative AI

During the preparation of this work, the authors used Perplexity in order to: Grammar and spelling check. After using these tool, the authors reviewed and edited the content as needed and takes full responsibility for the publication's content.

References

[1] Grishman, R. (2015). Information extraction. IEEE Intelligent Systems, 30(5), 8-15.

[2] Mai, T. T., Tran, Q. L., Tran, L. D., Ninh, T., Dang-Nguyen, D. T., & Gurrin, C. (2024, May). The First ACM Workshop on AI-Powered Question Answering Systems for Multimedia. In Proceedings of the 2024 International Conference on Multimedia Retrieval (pp. 1328-1329).

[3] Chowdhary, K., & Chowdhary, K. R. (2020). Natural language processing. Fundamentals of artificial intelligence, 603-649.

[4] Moreo, A., Esuli, A., & Sebastiani, F. (2016, July). Distributional random oversampling for imbalanced text classification. In Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval (pp. 805-808).

[5] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. Journal of artificial intelligence research, 16, 321-357.

[6] Mendes, P. N., Jakob, M., García-Silva, A., & Bizer, C. (2011, September). DBpedia spotlight: shedding light on the web of documents. In Proceedings of the 7th international conference on semantic systems (pp. 1-8).

[7] Kim, Y., Kim, B., & Lim, H. (2006, February). The index organizations for RDF and RDF schema. In 2006 8th International Conference Advanced Communication Technology (Vol. 3, pp. 4-pp). IEEE.

[8] Hogan, A., & Hogan, A. (2020). SPARQL query language. The Web of Data, 323-448.

[9] Sun, P., Yang, X., Zhao, X., & Wang, Z. (2018, November). An overview of named entity recognition. In 2018 International Conference on Asian Language Processing (IALP) (pp. 273-278). IEEE.

[10] Bouarroudj, W., Boufaida, Z. & Bellatreche, L. Named entity disambiguation in short texts over knowledge graphs. Knowl Inf Syst 64, 325–351 (2022).

[11] Bouarroudj, W., Boufaida, Z., Bellatreche, L. (2019). WeLink: A Named Entity Disambiguation Approach for a QAS over Knowledge Bases. In: Cuzzocrea, A., Greco, S., Larsen, H., Saccà, D., Andreasen, T., Christiansen, H. (eds) Flexible Query Answering Systems. FQAS 2019. Lecture Notes in Computer Science(), vol 11529. Springer, Cham.

[12] Jiang, E., Toh, E., Molina, A., Olson, K., Kayacik, C., Donsbach, A., ... & Terry, M. (2022, April). Discovering the syntax and strategies of natural language programming with generative language models. In Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (pp. 1-19).

[13] Kovács, G. (2019). An empirical comparison and evaluation of minority oversampling techniques on a large number of imbalanced datasets. Applied Soft Computing, 83, 105662.

[14] Mariani, G., Scheidegger, F., Istrate, R., Bekas, C., & Malossi, C. (2018). Bagan: Data augmentation with balancing gan. arXiv preprint arXiv:1803.09655.

[15] Mirza, M. (2014). Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784.

[16] Odena, A., Olah, C., & Shlens, J. (2017, July). Conditional image synthesis with auxiliary classifier gans. In International conference on machine learning (pp. 2642-2651). PMLR.

[17] Mullick, S. S., Datta, S., & Das, S. (2019). Generative adversarial minority oversampling. In Proceedings of the IEEE/CVF international conference on computer vision (pp. 1695-1704).

[18] Xu, L., Skoularidou, M., Cuesta-Infante, A., & Veeramachaneni, K. (2019). Modeling tabular data using conditional gan. Advances in neural information processing systems, 32.

[19] Belbekri, A., Benchikha, F., Slimani, Y., & Marir, N. SocialNER2. 0: A comprehensive dataset for enhancing named entity recognition in short human-produced text. Intelligent Data Analysis 28(3), 841 – 865 (2024).

[20] Bernier-Colborne, G., & Vajjala, S. (2024). Annotation Errors and NER: A Study with OntoNotes 5.0. arXiv preprint arXiv:2406.19172.

[21] Sang, E. F., & De Meulder, F. (2003). Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. arXiv preprint cs/0306050.