Srinivasan, Vardhini, and Power, Aurelia (2022) Character extraction and character type identification from summarised story plots. Journal of Computer-Assisted Linguistic Research 6: 19-41. ISSN: 2530-9455.

Character Extraction and Character Type Identification from Summarised Story Plots

Vardhini Srinivasan*, Aurelia Power
Technological University Dublin, Ireland
*Corresponding author: vardhini.srinivasan@gmail.com / B00124687@mytudublin.ie

Received: 10 June 2022 / Accepted: 14 September 2022 / Published: 25 November 2022

Abstract

Identifying the characters from free-form text and understanding the roles and relationships between them is an evolving area of research. They have a wide range of applications, from summarising narrations to understanding the social network from social media tweets, which can help in automation and improve the experience of AI systems like chatbots and much more. The aim of this research is twofold. Firstly, we aim to develop an effective method of extracting characters from a story summary, to develop a set of relevant features, then, using supervised learning algorithms, to identify the character types. Secondly, we aim to examine the efficacy of unsupervised learning algorithms in type identification, as it is challenging to find a dataset with a predetermined list of characters, roles, and relationships that are essential for supervised learning. To do so, we used summary plots of fictional stories to experiment and evaluate our approach. Our character extraction approach successfully improved on the performance reported by existing work, with an average F1-score of 0.86. Supervised learning algorithms successfully identified the character types and achieved an overall average F1-score of 0.94. However, the clustering algorithms identified more than three clusters, indicating that more research is needed to improve their efficacy.

Keywords: character extraction, character type identification, coreference resolution, classification, clustering.

DOI: HTTPS://DOI.ORG/10.4995/JCLR.2022.17835
PUBLISHER: UNIVERSITAT POLITÈCNICA DE VALÈNCIA.

1. Introduction

Automatic extraction and identification of characters and personas types from free-form texts has recently received increased attention due to the wide range of applications, including tasks such as providing a synopsis of legal cases, identifying victims from legal case notes, identifying relationships, building social networks between the characters, and analysing social media texts to identify victims and bullies (Labatut and Bost 2020; Vala et al. 2015). In this paper, we describe a novel heuristic-based approach to extracting characters. We attempt to extract characters using a five-step methodology, followed by feature set creation which employs NLP techniques and heuristic rules. We then identify the character types going beyond the traditional binary set of types – characters vs non-characters. As such, we define the set of character types to include three labels: protagonist, antagonist, and supporting character, which, to the best of our knowledge, has not been attempted before. We also compare the performances of supervised and unsupervised learning algorithms in identifying the character type.

The remainder of the paper is structured as follows: Section 2 summarises the research carried out in the areas of character extraction and character type identification, respectively. Section 3 describes the methodology employed here to extract characters, create the feature set, and carry out the classification and clustering experiments aimed at identifying the character types. Section 4 reports and discusses the performance of character extraction and the classification experiments in terms of precision, recall, and F1 score, while the results obtained from the clustering experiments in terms of homogeneity, completeness, and V-measure. We conclude our paper in section 5, identifying our contributions and providing directions for future research.

2. Related work

Characters are nominal entities that are vital to a story, contributing significantly to the flow of the plot through the actions they perform (Jahan et al. 2019). They are animate entities, irrespective of their nature, in other words, animals, humans, and anthropomorphic objects can all be considered characters, as long as they actively participate in the plot in a meaningful way. Grammatically, characters can be represented using proper nouns, pronouns, as well as various other anaphoric and cataphoric noun phrases. Characters can be categorised into distinct types based on the level of contribution to the plot in conjunction with judgments against a set of pre-established ethical values (Talib 2010). For instance, characters can be protagonists or antagonists, both playing central roles, however, differing in terms of ethical and moral status: protagonists generally display favourable qualities, such as bravery, loyalty, kindness, etc., while antagonists have personality traits such as impulsivity, self-interest, etc.

Character extraction and character type identification are interlinked tasks, the former being a prerequisite to the latter. The task of extraction aims to identify and extract the characters from a free text (Labatut and Bost 2019). On the other hand, character type identification focuses on

determining the types of extracted characters, such as protagonist, antagonist, or support characters (Talib et al. 2010).

Two approaches to character extraction were adopted by most studies: heuristic-based approaches which rely on NLP techniques and grammatical rules, and hybrid approaches which, in addition to heuristics, also utilize ML algorithms. Irrespective of the techniques used, character extraction requires structural-level and context-level narrative information (Labatut and Bost 2019). This type of information can be extracted via NLP mechanisms such as named entity recognition (NER) - the task of detecting all regularly named entities such as the name of locations, organizations, persons, or unique entities like the name of chemical components (Finkel et al. 2005), and named entity linking (NEL) – the task of identifying mentions from a text and linking them to the corresponding entity they name that can be found in an external knowledge base (Hachey et al. 2013) For example, coreference[1] resolution, which is the process of identifying all instances in a piece of text that refer to the same entity (Manning et al. 2014), plays a crucial role in extracting the characters and the variant names (Labatut and Bost 2019; Liang and Wu, 2004; Vala et al. 2015), as a particular character can be referenced in multiple ways, such as using gender based pronouns (e.g., *she* is used to refer to a female character), honorifics (for instance, Dr is used to refer to characters who hold a doctoral degree or a medical degree), nicknames (e.g., *Romeo* from *The Adventures of Pinocchio* was nicknamed *Candlewick* because he is very tall and thin), as well as name variants (for instance, some characters may be referred to by using their full name, or only the first name, or only the last name). In extracting nominal entities, heuristic-based approaches have relied on grammatical information such as parts of speech or gender/number agreement, as well as external knowledge, such as the WordNet lexical database, to confirm the animacy of these nominal entities (Chen and Choi 2016; Jahan et al. 2018; Liang and Wu 2004; Valls-Vargas et al. 2014), while others have used additional ML algorithms (Agarwal and Rambow 2010; Calix et al. 2013; Jahan et al. 2019, 2020; Valls-Vargas et al. 2014) to embed various types of action-based features and lexical features in the classification models.

Character type identification has also been generally approached in two ways. The first approach, which dominates the character type identification landscape, uses supervised learning (Chaturvedi et al. 2016; Fernandez et al. 2015) to classify the relationship between two characters as cooperative and non-cooperative, or to classify characters as protagonists or other types of characters. The second approach, which is seldomly employed, uses unsupervised learning to group similar characters together (Bamman et al. 2013; Jung et al. 2013). While the supervised approach has exclusively focused on binary classification (Chaturvedi et al. 2016; Fernandez et al. 2015), the unsupervised approach has focused on grammatical dependencies (Bamman et al. 2013; Noah et al. 2013) or on emotional/sentiment similarities (Jung et al. 2013). Moreover, existing research on using an unsupervised approach failed to reliably identify the natural groupings of characters. Across both approaches, it is clear that existing research neglected to address identifying more than one character type, with only one paper attempting

[1] Like many other authors, such as Sukthanker et al. (2020), we use the term coreference resolution to also refer to anaphoric and cataphoric resolution.

to identify protagonists, antagonists, and tritagonists (Jung et al. 2013), but limited to unsupervised learning. Additionally, only one study (Bamman et al. 2013) attempted to perform clustering in addition to classification by grouping characters based on similarity measures, however, the authors did not provide any information on how the characters were extracted.

3. Methodology

Our study aimed to develop a novel heuristic-based approach to extract characters from summarised story plots using a ternary system to label the characters as protagonist, antagonist, and support. The output of the character extraction stage is then used to evaluate both supervised and unsupervised approaches to identifying the type of characters we extracted. To achieve this, we propose four main stages: (1) data acquisition and dataset creation on which our approach was developed and evaluated, (2) character extraction which was designed to extract all the characters from each story plot, (3) feature set creation which was developed to identify relevant features along which character vectors are generated, and (4) character type identification aimed at using supervised and unsupervised approaches to identify the three types of characters: protagonists, antagonists, and support characters.

3.1. Data Acquisition and Dataset Creation

We obtained summary plots of 20 fictional stories with a total of 218 characters using Web-scraping scripts from SparkNotes.com. We stored the extracted details in two consumable comma-separated files (CSV) which had the titles and the corresponding summary plots, and the titles and the corresponding characters list, respectively. The length of the summary plots varied from 3000 to 8000 letters, while the number of sentences in each plot varied from 25 to 74, as shown in Table 1.

Title	Number of Characters	Number of Sentences	Summary Text Length
To Kill A Mockingbird	11	39	4372
The Adventures of Tom Sawyer	14	44	5634
As You Like It	15	45	6903
The Bean Trees	14	70	7691
A Doll's House	7	74	5405
Educated	7	42	4215
Fool For Love	5	45	4456
Giants in the Earth	10	53	4059
Across Five Aprils	13	37	4480
Great Expectations	15	49	5866
The Kite Runner	12	69	6253
Twelfth Night	10	38	4971
The Secret Garden	9	45	5512
King Lear	12	25	2814

The Crucible	12	45	4865
Frankenstein	7	52	5721
The Giver	6	49	6463
The Moonstone	14	39	5122
Pride and Prejudice	15	56	6573
A View from the Bridge	9	42	3593

TABLE 1: DATASET DETAILS

The character type distribution from the data set is shown in Figure 1. The class imbalance seen is natural for the stories, with only few protagonists and antagonist characters.

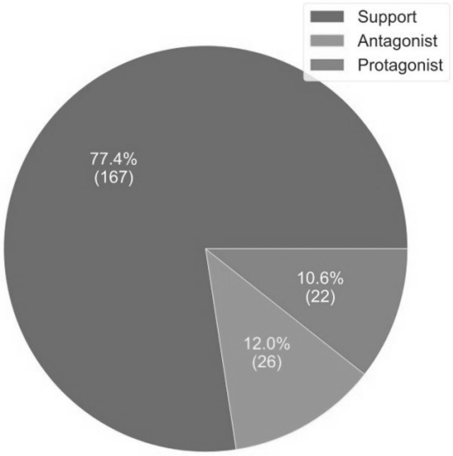

FIGURE 1: CHARACTER TYPES DISTRIBUTION

3.2. Character Extraction

We designed a five-step approach to extract the characters from the summary plot.

(1) Data Preprocessing and Annotation: we first removed non-ASCII characters that the webscraping script introduced in the summary plot such as carriage returns, tabs, and page breaks, as they acted as noise and did not provide any predictive value. We then sentence-tokenised each plot and annotated it using the StanfordNLP tool. The following annotators were used: (a) parts of speech tagging to identify the parts of speech of sentence constituents, such as nouns, verbs, etc., (b) named entity recognition to identify any applicable entity, such as PERSON, TITLE, etc., (c) dependency parsing to analyse the relationships between sentence constituents and identify their role, such as root/verb, nominal subject, object, etc., (d) coreference chaining and resolution to identify and resolve any corefering expressions such as pronouns to the same nominal entity, and (e) Subject-Verb-Object triplet extraction. We then reconstructed the summary plot into paragraphs and performed the sentence tokenisation and annotation a second time to provide the

annotator with smaller logical units so that coreferences within a sentence can be more efficiently and more accurately identified.

(2) Proper Nominal Entities (PNE) Identification: using the POS tags from the annotated text, we identified all proper nouns (NNP) and joined those proper nouns that represent a single nominal entity by checking whether they occur consecutively; for instance, in the sentence 'Tom Sawyer goes to school regularly.', we join 'Tom' and 'Sawyer'. We then obtain all personal pronouns and identify gender using the coreference chain as it contains the pronouns of a nominal entity (reference head) and other nominal references; for example, from the sentences 'Tom Sawyer goes to school regularly. Tom and his classmate Jim often go to an abandoned house to play.', we retrieve [Tom Sawyer, Tom, his], as 'his' indicates the gender of the nominal entity to which it refers. This step was necessary as names alone cannot guarantee that gender can be correctly identified, since many names can be used for both genders.

(3) Gender Confirmation: we confirmed the gender with three heuristic rules. (a) First, based on the count of pronouns in the coreference chain, we confirmed the correct gender. For instance, the following paragraph, 'Scout Finch lived in the town of Maycomb. She lived with her father and her brother', produces a coreference chain containing the following mentions: [Scout Finch, She, her, her], with one count of 'male' for 'Scout Finch' because 'Scout' typically denotes a male (despite that it represents a female character in this context) and three counts of 'female' provided by the three pronouns and, as the count of 'female' is greater than that of 'male', the gender of 'Scout Finch' was set to 'female'. (b) Secondly, we retrieve the Subject-Verb-Object (S-V-O) triplets from the annotator and extract the gender of the objective pronouns if they are present. We then confirm the gender of the entity by identifying the gender of the objective pronoun. For instance, in the following paragraph, 'Sam and Sarah were playing together. Sarah fought with him', for the second sentence, the S-V-O returned was 'Sarah' as the subject, 'fought' as the verb, and 'him' as the object; we then confirmed 'Sam' to be 'male' based on the fact that its coreference chain contains 'him' which is 'male'. (c) Finally, based on the gender of the honorific titles, we align the gender one more time. For example, in the sentence 'Mrs. Smith went to the shop and bought ice cream', the PNE has the honorific title Mrs. which is of the 'female' gender. Thus, we confirm if the gender identified by the previous steps matches the honorific title.

(4) Alias Resolution: in this step, we identify the variant names using an approach inspired by the work of Vala et al. (2015). To do so, we first created a character graph with all the identified PNEs as nodes and generated edges between two PNEs if they were mentioned in the same coreference chain. As an example, Figure 2 shows the initial PNEs graph generated from the story plot 'To Kill A Mockingbird' where not all nodes are connected.

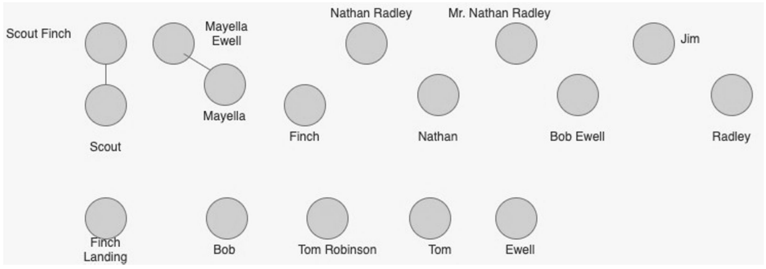

FIGURE 2: 'TO KILL A MOCKINGBIRD' PNES GRAPH BEFORE HEURISTIC RULES ARE APPLIED

We then applied the following heuristic rules to all the identified PNEs to determine if they were alias or name variants of the same PNE and, thus, further connect PNEs: (a) if the first names or sub-string of the first names match and the last names match, for example, 'Tom Robinson' and 'Toms Robinson', (b) if the last names or the first names match and the gender of honorifics (if any) matches, e.g., 'Mr. Nathan Radley'and 'Mr. Radley', or 'Lady Catherine' and 'Lady Catherine de Bourgh', and (c) if the full names match and only one has an honorific, e.g., 'Mr. Nathan Radley'and 'Nathan Radley'. Alternatively, we avoided name conflicts that denote different PNES, thus, not connecting them or even eliminating edges between two PNEs using the following heuristic rules: (a) if PNEs share the last name, but not the first name, e.g., 'Mayella Ewell' and 'Bob Ewell', (b) if PNEs share the last name, but have different honorifics belonging to different genders, for example, 'Mr. Bennet' and 'Mrs. Bennet', and (c) if PNEs have a last name only, e.g., 'Ewell', or are preceded by a determiner, e.g., 'The Bennets'. As shown in Figure 3, the result of applying these rules leads to more edges being generated between nodes, while at the same time avoiding names.

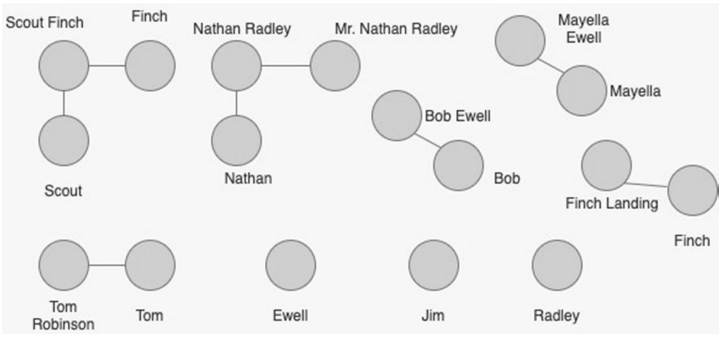

FIGURE 3: 'TO KILL A MOCKINGBIRD' PNES GRAPH AFTER ALL HEURISTIC RULES ARE APPLIED

At the end of this step, we get a list of PNEs denoting different characters, with their alias names captured as a feature, and set the gender of the head PNE as the gender of the majority of the alias PNEs. Once we added the alias list, we removed them from the data frame. For example: 'Scout Finch' had the gender as male, 'Scout' had it as female, 'Finch' had it as female. We added 'Scout' and 'Finch' as alias names to 'Scout Finch', and set the gender to female.

(5) Elimination of non-characters: we applied a final set of heuristics to eliminate any non-characters from the list. Thus, a PNE was removed if at least three of the following four conditions below were met: (a) the PNE had not appeared as a subject in any sentence, (b) the PNE was not a person's name or person's title, (c) the WordNet lexical database did not recognize the PNE as a noun, and (d) the PNE did not have any alias PNEs associated with it. For example, the PNE 'Finch Landing' was removed because the following three conditions were met: it did not appear in any sentence as a subject; it is a location and not a person's name or title; and the WordNet did not recognize the PNE as a noun.

For comparison, we designed a baseline approach using CoreNLP basic annotation process: we annotated each summary plot after breaking them into paragraphs and, from the resulting annotated sentences, we extracted the PNEs and their title based on whether the NER tag was equal to PERSON; the alias list was left empty so as to use the same logic to compare the effectiveness in finding alias names.

3.3. Feature Set Creation

The purpose of this stage was to identify and extract a set of relevant features to be used as independent variables during the character type identification stage. To determine these features, we first performed coreference and alias resolution, and replaced all the pronouns and other nominal references of a PNE with the PNE itself. For example, 'Tom Sawyer lived in a village with his family' would be changed to 'Tom Sawyer lived in a village with Tom Sawyer family'. We then re-annotated the text using the same annotators used during the preprocessing and extracted the following features for each PNE:

(1) Basic features: the PNE name, the title of the story in which it appears, its gender, and its alias list.
(2) Grammatical features: the list of POS tags associated with the PNE and two Boolean features indicating whether a PNE appeared as a subject and as an object, respectively.
(3) Contextual features: the list of verb phrases representing actions performed by the PNE as an agent, the list of verb phrases representing actions performed on the PNE as a patient, a list of attributes (adjectival and adverbial modifier phrases) that describe the PNE, the list of NER tags associated with the PNE; for each of these lists of phrases we have computed the sentiment scores and added them to the feature set, where the sentiment score was computed using the SentWordNet (Baccianella et al. 2010).
(4) Statistical features were derived as counts or relative frequencies and included the following: the number of times the PNE was mentioned, the number of times the PNE acted as a subject, the number of times the PNE acted as an object, and the normalised coreference score which we calculated using the following formula (Jahan et al. 2020):

(1) $$z = \frac{(x-\mu)}{\sigma}$$

where x is the PNE's chain length, μ is the chain length mean, and σ is the chain length standard deviation.

In total, we extracted 26 relevant features. Table 2 summarises these features.

Feature Name	Type	Description
Title	Text	Title of the story
PNE	Text	Proper nominal entity or character name
POS	Text	Part of Speech Tag
NER	Text	NER type
Gender	Text	Gender (Male, Female, Unknown)
NumOfMentions	Numerical	Number of times a PNE was mentioned
Alias	List of Text	List of Alias names
Agent Phrases	List of Text	List of Agent Phrases
Patient Phrases	List of Text	List of Patient Phrases
Attribute Phrases	List of Text	List of Attribute Phrases
NumOfNSubj	Numerical	Number of times PNE had been a nominal subject
CoreferenceScore	Numerical	Coreference Score
TPSubjectCount	Numerical	Number of times PNE had been a subject in S-V-O
TPObjectCount	Numerical	Number of times PNE had been an object in S-V-O
SVO AgentPhrases	List of Text	Agent Phrases from S-V-O
SVO PatientPhrases	List of Text	Patient Phrases from S-V-O
AgentSentiScore	Numerical	Sentiment score of agent phrases
PatientSentiScore	Numerical	Sentiment score of patient phrases
AttributeSentiScore	Numerical	Sentiment score of attribute phrases
SVO AgentSentiScore	Numerical	Sentiment score of S-V-O agent phrases
SVO PatientSentiScore	Numerical	Sentiment score of S-V-O patient phrases
NPIsAgent	Boolean	If the PNE had been an agent in any sentence
NPIsPatient	Boolean	If the PNE had been a patient in any sentence
NPIsSubject	Boolean	If the PNE had been a subject in any sentence
NPIsObject	Boolean	If the PNE had been an object in any sentence

TABLE 2: FEATURE SET

3.4. Character Type Identification

The purpose of this stage was to apply supervised and unsupervised algorithms to identify the character types. To achieve this, we first performed dataset preparation to get the dataset ready for the machine learning (ML) algorithms during this stage, and then addressed class imbalance which was evident in Figure 1. We then applied several algorithms, both supervised and unsupervised, to identify the character type and attempted to further improve performance by applying hyperparameter tuning. These steps are described below:

(1) Data Preparation: we first vectorised the dataset along with the features we identified in 3.3 to which we added the label corresponding to each character using the following class labels set: <protagonist, antagonist, support>. We also performed the following numerical data transformations so that ML algorithms can be applied: (a) the values of ordinal and categorical features were converted to discrete numerical values, and (b) the values of features that represented list data structures were replaced by their length denoting the number of items in the list. Note that we applied the classification algorithms after this step to obtain a baseline level of performance, before we evaluated the cumulative contribution of class imbalance techniques and hyperparameter tuning techniques.

(2) Addressing Class Imbalance: we adopted the Synthetic Minority Oversampling Technique (SMOTE) methodology of oversampling to address the imbalance since our dataset is relatively small making it unsuitable for undersampling. SMOTE adds synthetic tuples that are closer to the positive tuples to increase the sample size (Ceri et al. 2003).

(3) Applying Supervised Learning Algorithms: we selected algorithms that can perform well with small training sets (Ceri et al. 2003), such as K-Nearest-Neighbour (KNN), and two ensemble algorithms – Gradient Boosting Classifier (GBC) and Random Forest Classifier (RFC). To improve their performance, we further applied hyperparameter tuning to find the optimal parameters set using a random search: (a) in the case of KNN, we varied the number of neighbouring samples, the weighting scheme which indicated whether neighbouring samples should be treated equally, or whether closer points should be given more importance, and the metric used to measure distances between samples; (b) for GBC, we experimented with the number of estimators which indicates the number of models built, the maximum number of nodes allowed in each model, the minimum number of samples required to split a node, and the maximum number of features to consider at each split; finally, (c) in the case of RFC we experimented with the same parameters as for GBC, as well as an additional one which indicates whether to consider the entire dataset or portion of the dataset when building the tree.

(4) Applying Unsupervised Learning Algorithms: we used K-Means, Agglomerative Clustering, and Density-Based Spatial Clustering (DBScan) to cluster the characters based on the type. Like in the case of supervised algorithms, we have attempted to further improve the performance of clustering algorithms by experimenting with various parameters: (a) for K-Means we have investigated a various number of clusters, as well as the type of initialisation which indicates whether the initial centroids are randomly chosen or as far apart as possible; (b) in the case of agglomerative clustering, we experimented with the number of clusters, as well as the linkage scheme which indicates the distance criterion for merging clusters; and (c) for DBScan, we varied the maximum distance allowed between samples to be considered neighbours, and the minimum number of samples required for a core point.

3.5. Performance Evaluation

We measured the performance of the character extraction task using precision, recall, and F1-score, where precision is a measure of exactness, recall is a measure of completeness, and the F1-score is the harmonic mean of both precision and recall (Grandini et al. 2020). They were

computed for two classes – character and non-character – based on the number of true and false positives, and true and false negatives, according to the following formulas:

$$(2) \quad Precision = \frac{True\ Positives}{True\ Postives + False\ Positives}$$

$$(3) \quad Recall = \frac{True\ Positives}{True\ Postives + False\ Negatives}$$

$$(4) \quad F1 - score = 2\ X\ \frac{Precision\ X\ Recall}{Precision + Recall}$$

In the context of character extraction, we define true positives as those instances that were correctly identified as characters, whereas false positives are those instances that were incorrectly identified as characters but were not in fact characters. On the other hand, we define true negatives as those instances that were correctly identified as non-characters, while false negatives are those instances that the approach failed to identify them as a character despite that they were in fact characters.

To measure the performance of the classification algorithms, we also used precision, recall, and F-score, however, they were computed as macro averages (Grandini et al. 2020) across the three-character types, that is, the scores for each character type were accumulated and then divided by the number of classes or unique character types (protagonist, antagonist, and support), as demonstrated by the formulas below:

$$(5) \quad Macro\ Average\ Precision = \frac{\sum_{i=1}^{n} Precision_i}{n}$$

$$(6) \quad Macro\ Average\ Recall = \frac{\sum_{i=1}^{n} Recall_i}{n}$$

$$(7) \quad Macro\ Average\ F - Score\ = \frac{\sum_{i=1}^{n} F{-}Sco\ \ _i}{n}$$

where *n* represents the total number of classes (here, *n*=3) and *i* represents a given class.

To measure the performance of clustering algorithms, we used homogeneity which indicates whether samples belong to one single class, completeness which indicates whether samples belonging to a single class belong to a single cluster, and the V-measure which denotes the harmonic mean of homogeneity and completeness scores (Rosenberg and Hirschberg 2007). The formulas describing these metrics are listed below:

$$(8) \quad Homogeneity = 1 - \frac{H(C|K)}{H(C)}$$

where $H(C|K)$ is the conditional entropy of the classes given the cluster assignments, while $H(C)$ is the entropy of the classes.

$$(9) \quad Completeness = 1 - \frac{H(K|C)}{H(K)}$$

where $H(K|C)$ is the conditional entropy of the clusters given the classes, while $H(K)$ is the entropy of the clusters.

$$(10)\quad V-measure = 2\,X\,\frac{Homogeneity\ X\ Completeness}{Homogeneity + Completeness}$$

4. Results and Analysis

We implemented our character extraction approach using Python packages and libraries for NLP and ML, such as the Stanford CoreNLP library (Manning et al. 2014). We extracted 20 summary plots of fictional stories which contained 218 characters. We have also prepared a hand-labelled gold standard dataset used to compute the performances of two approaches: the character extraction approach based on basic CoreNLP elements which served as the baseline approach, and our five-step approach. The results are shown in Table 3.

Story Title	Precision		Recall		F1 Score	
	Basic CoreNLP	Our Approach	Basic CoreNLP	Our Approach	Basic CoreNLP	Our Approach
To Kill A Mocking Bird	0.6923	0.8462	0.8182	1	0.75	0.9167
The Adventures of Tom Sawyer	0.4444	0.9333	0.5714	1	0.5	0.9655
As You Like It	0.6316	1	0.8	0.8667	0.7059	0.9286
The Bean Trees	0.4348	0.7143	0.6667	1	0.5263	0.8333
A Doll's House	0.3636	0.7	0.5714	1	0.4444	0.8235
Educated	0.8571	0.6	0.8571	0.8571	0.8571	0.7059
Fool For Love	1	1	0.6	1	0.75	1
Giants in the Earth	0.7273	0.9	0.8	0.9	0.7619	0.9
Across Five Aprils	0.6667	0.8	0.7692	0.9231	0.7143	0.8572
Great Expectations	0.6	0.8824	0.6	1	0.6	0.9375
The Kite Runner	0.7692	0.9231	0.8333	1	0.8	0.96
Twelfth Night	0.5714	0.8333	0.8	1	0.6666	0.9091
The Secret Garden	0.25	0.6923	0.3333	1	0.2857	0.8182
King Lear	0.8571	1	0.5	0.6667	0.6316	0.8
The Crucible	0.5625	1	0.75	1	0.6429	1
Frankenstein	0.3636	0.75	0.5714	0.8571	0.4444	0.8
The Giver	1	0.5	0.5	0.6667	0.6667	0.5714
The Moonstone	0.3636	0.7368	0.5714	1	0.4444	0.8485
Pride and Prejudice	0.4762	0.75	0.6667	1	0.5556	0.8571
A View from the Bridge	0.8889	0.7273	0.8889	0.8889	0.8889	0.8

TABLE 3: RESULTS OF CHARACTER EXTRACTION EXPERIMENTS

Overall, our approach outperformed the baseline CoreNLP approach, with our approach yielding an average F1-score of 0.86, while the baseline average F1-score was 0.63. Furthermore, a higher average recall value of 0.93 was achieved by our approach, while the baseline achieved an average recall value of 0.67, indicating that our novel five-step approach identified most of the characters in the gold-standard dataset and did not miss many. Similarly, on average, our approach outperformed the baseline in terms of precision, with an average precision of 0.74 and 0.63, respectively. However, in the case of two stories – 'Educated' and 'The Giver' – the baseline approach was more effective, the lower precision score associated with our approach being due to the higher number of false positives: in the case of 'Educated', one false positive was yielded by the baseline as opposed to four false positives yielded by our approach, while

in the case of 'The Giver', the baseline approach produced no false positives, whereas our approach produced four false positives.

In total, our approach identified 240 support characters, 19 antagonists, and 26 protagonists, indicating that, while it did not miss identifying many characters, it had some issues with finding all the name variants, as highlighted by the findings of our error analysis, across all 20 summarised story plots:

(1) In total, our approach produced 48 false positives, that is, it identified 48 PNEs as characters that were not in fact characters. For example, 'House' and 'Nursing Center' from 'The Giver' were extracted as characters, despite verifying their animacy, which may indicate issues related to the last step in our approach – elimination of non-characters.

(2) Overall, our approach had 32 PNEs duplicated, which indicates that the alias resolution procedure did not capture all PNEs under the same parent node, possibly, due to two factors: (a) issues associated with last name resolution: when a character was referred with last name alone and had the full name or different honorific mentioned in other references, they were not identified as an alias, but rather as separate (but duplicate) PNEs; for example, in 'The Moonstone', one of the characters was referred to as 'Sergeant Cuff' as well as 'Cuff', however, the approach extracted two different characters, and (b) issues associated with gender resolution: when a character had different honorifics, gender resolution was inaccurate and they were identified as different characters, thus leading to duplicate PNEs, although this issue appeared in one case in the 'Moonstone', wherein 'Dr. Candy', 'Mr. Candy', and 'Candy'.

Table 4 summarises the performance obtained from the supervised learning (classification) experiments. We applied ML algorithms to the feature set generated during the second stage of our methodology and evaluated them against the gold-standard dataset.

Model	Character Type	Baseline			SMOTE Applied			Best Fit Model		
		Precision	Recall	F1 Score	Precision	Recall	F1 Score	Precision	Recall	F1 Score
K Nearest Neighbor	Protagonist	0	0	0	0.84	1	0.91	0.84	1	0.91
	Antagonist	0.6	0.3	0.4	0.86	0.95	0.9	0.88	1	0.94
	Support	0.77	0.96	0.86	0.9	0.56	0.69	1	0.59	0.74
Gradient Boosting Classifier	Protagonist	0	0	0	0.88	1	0.93	0.95	1	0.98
	Antagonist	0	0	0	0.9	1	0.95	1	1	1
	Support	0.75	0.98	0.85	1	0.68	0.81	1	0.94	0.97
Random Forest Classifier	Protagonist	0	0	0	0.95	1	0.98	0.98	1	0.99
	Antagonist	0.8	0.4	0.53	0.92	1	0.96	0.98	1	0.99
	Support	0.79	0.98	0.88	1	0.82	0.9	1	0.94	0.97

TABLE 4: RESULT OF CLASSIFICATION EXPERIMENTS

Overall, the baseline produced relatively low scores, with an average F1-score of 0.39, with lowest of 0.28 F1-score for GBC. As can be seen, the baseline approach failed to capture any of the protagonist characters, irrespective of the classification algorithm used. Additionally, KNN and RFC produced relatively low scores for the antagonists; the GBC algorithm failed to capture any antagonists. On the other hand, the baseline performance for the support characters is

relatively high, with a recall above 0.95 for all algorithms. This is explained by the severe class imbalance depicted in Figure 1 and further confirmed by the dramatic improvement of more than 85% for predicting protagonists and antagonists once SMOTE was applied to address the class imbalance, representing all character types equally with 240 samples each, although KNN showed a decrease in recall for support characters. We further improved performance by applying hyper-parameter tuning, achieving an average F1-score of 0.94.

In terms of algorithms, while KNN and RFC performed better on the baseline, GBC and RFC produced the best final results, both with an average F1 score of 0.98, although KNN also produced dramatic increases, with an average F1 score of 0.86. This boost in performance is mainly explained by addressing the class imbalance, but also by choosing an optimal set of parameters applicable to each algorithm as follows: (a) KNN shown best performance when K-value was 5, and when the neighbouring data points were weighted using the 'distance' option, (b) best performance from the GBC was obtained with the number of classifiers set to 150, maximum depth set to 5, the minimum samples split set to 5, and the maximum features to consider set to auto, and, finally, (c) for RFC, the best performance was obtained with 100 classifiers, the bootstrap parameter set to false, the maximum depth set to 80, and the maximum features set to auto.

While classification experiments demonstrated great performance, the clustering algorithms did not perform as well when we attempted to cluster the samples into three clusters to represent the three classes: protagonist, antagonist, and support character. As shown in the Table 5, homogeneity, completeness, and V-measure values were low across all algorithms in the baseline approach, ranging from the lowest V-measure score of 0.04 for DBScan to the highest V-measure score of 0.18 for Agglomerative Clustering. Furthermore, although applying SMOTE did increase performance across all measures, it did not increase it to a meaningful level, with V-measure scores of 0.185 in the case of K-Means, 0.175 in the case of Agglomerative Clustering, and 0.473 in the case of DBScan, respectively. One notable exception is the DBScan which displays a marked improvement in the homogeneity score, from 0.046 to 0.96, representing an increase of more than 20 times.

Model	Baseline			SMOTE Applied		
	Homogeneity	Completeness	V-measure	Homogeneity	Completeness	V-measure
K-Means	0.1584	0.1752	0.1664	0.1545	0.229	0.1845
Agglomerative Clustering	0.1562	0.2021	0.1762	0.1411	0.2302	0.1749
DBScan	0.0463	0.0346	0.0396	0.9624	0.3136	0.473

TABLE 5: RESULTS OF CLUSTERING EXPERIMENTS WITH THREE CLUSTERS

The poor performance of clustering algorithms with three clusters indicates that a different number of clusters may be more suitable to avoid the presence of overlapping characters and outliers that the algorithms find difficult to group. In an attempt to find the optimum number of clusters, we run experiments with a various number of clusters, ranging from 3 to 14, and various parameter settings. Figure 4 shows the results of applying K-Means across various parameter settings and the number of clusters.

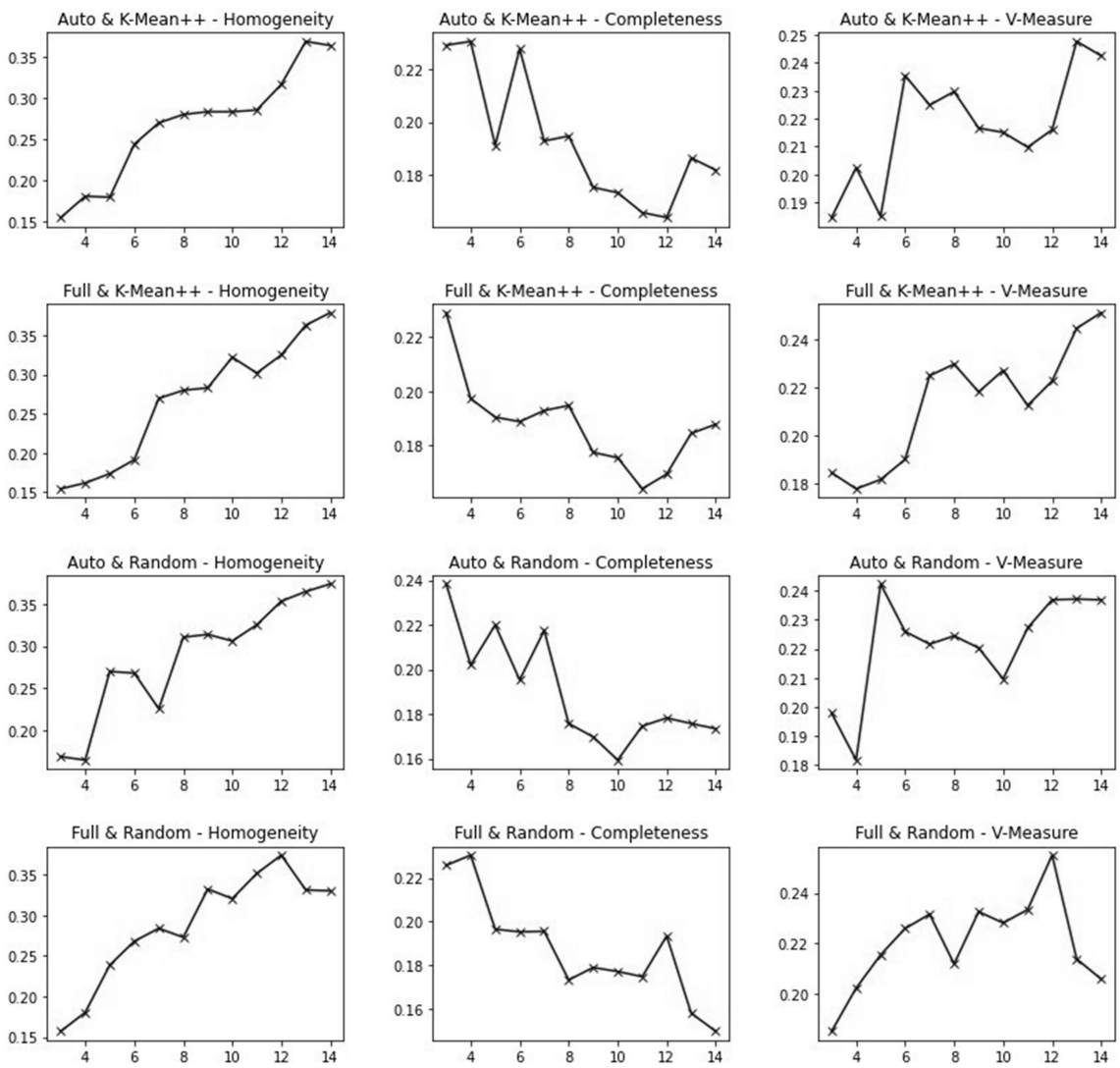

FIGURE 4: K-MEANS RESULTS FOR DIFFERENT NUMBERS OF CLUSTERS AND INITIALISATION SCHEMES

As can be seen, overall, the lowest V-measure scores were achieved in those cases where the number of clusters was set to 3, confirming our initial suspicion that three clusters may not be the optimal number for our data. On the other hand, the best V-measure scores were achieved in the case of 5, 12, 13, and 14 clusters, respectively. The results of the experiments for Agglomerative Clustering were similar in that, overall, the number of clusters set to 3 produced the lowest V-measure scores, while 10 clusters and 14 clusters led to the highest V-measure scores, respectively.

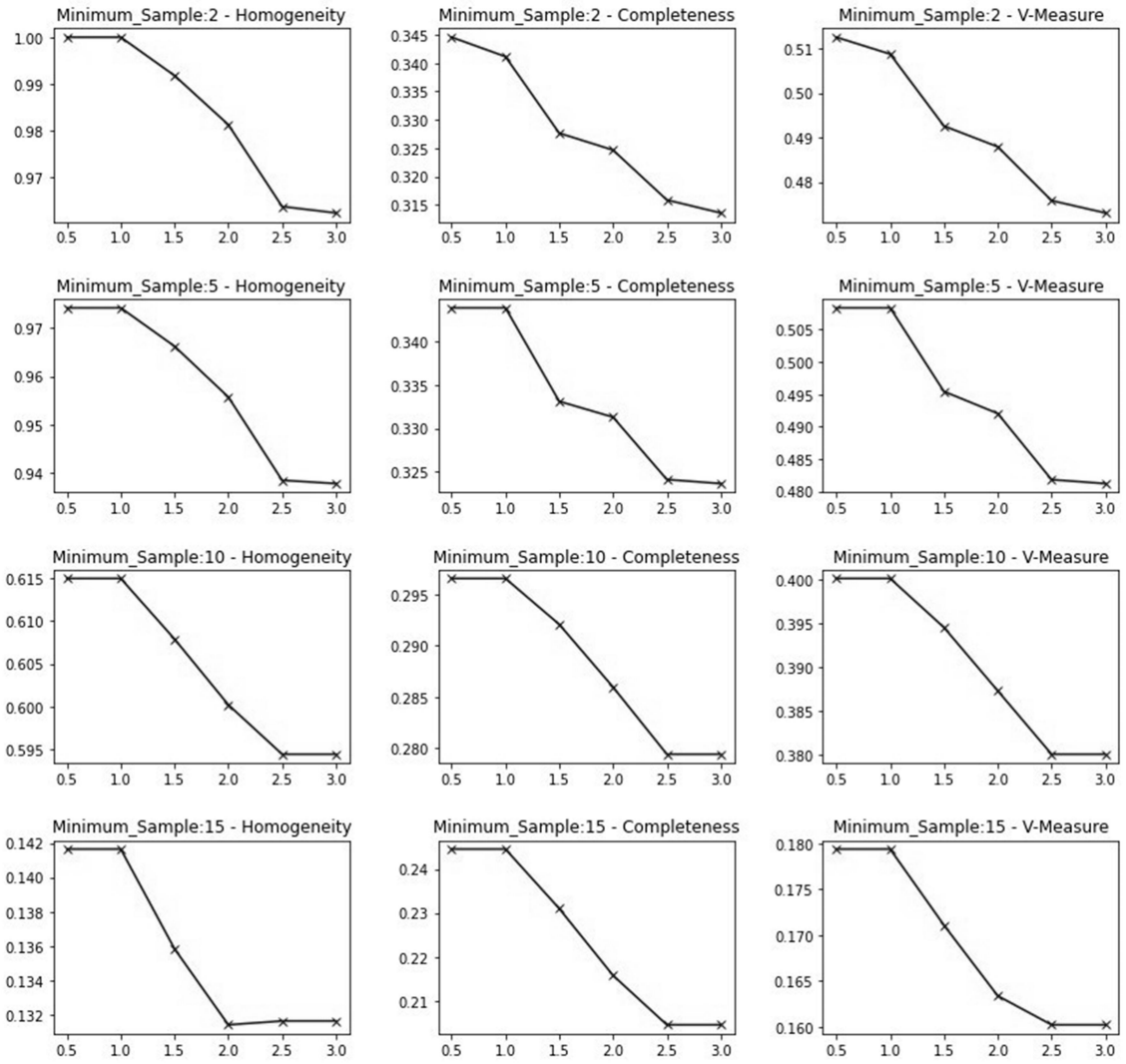

FIGURE 5: DBSCAN RESULTS FOR DIFFERENT NUMBERS OF EPS AND SAMPLES

In the case of DBScan, as shown in Figure 5, better results were consistently achieved with lower values of eps which represent the maximum distances between neighbours. In terms of the different number of minimum data points required as neighbouring points we experimented with, the best performance was achieved with a value of 2. Taken together, these results indicate that the DBScan performs better with a smaller number of neighbouring samples and lower values of distances between these neighbouring samples, thus with a greater number of clusters than three.

To get a better understanding of how K-Means and Agglomerative Clustering algorithms distributed the characters (data points) among clusters in those cases where performance reached the best results, we mapped the cluster numbers to the rows in the dataset and checked them against the label. Thus, we have done so using 12 clusters for K-Means, and 14 clusters for Agglomerative Clustering, respectively for both, the data set outputted from the character extraction stage, as well as its SMOTE upsampled version. The results are shown in Figures 6 and 7, respectively.

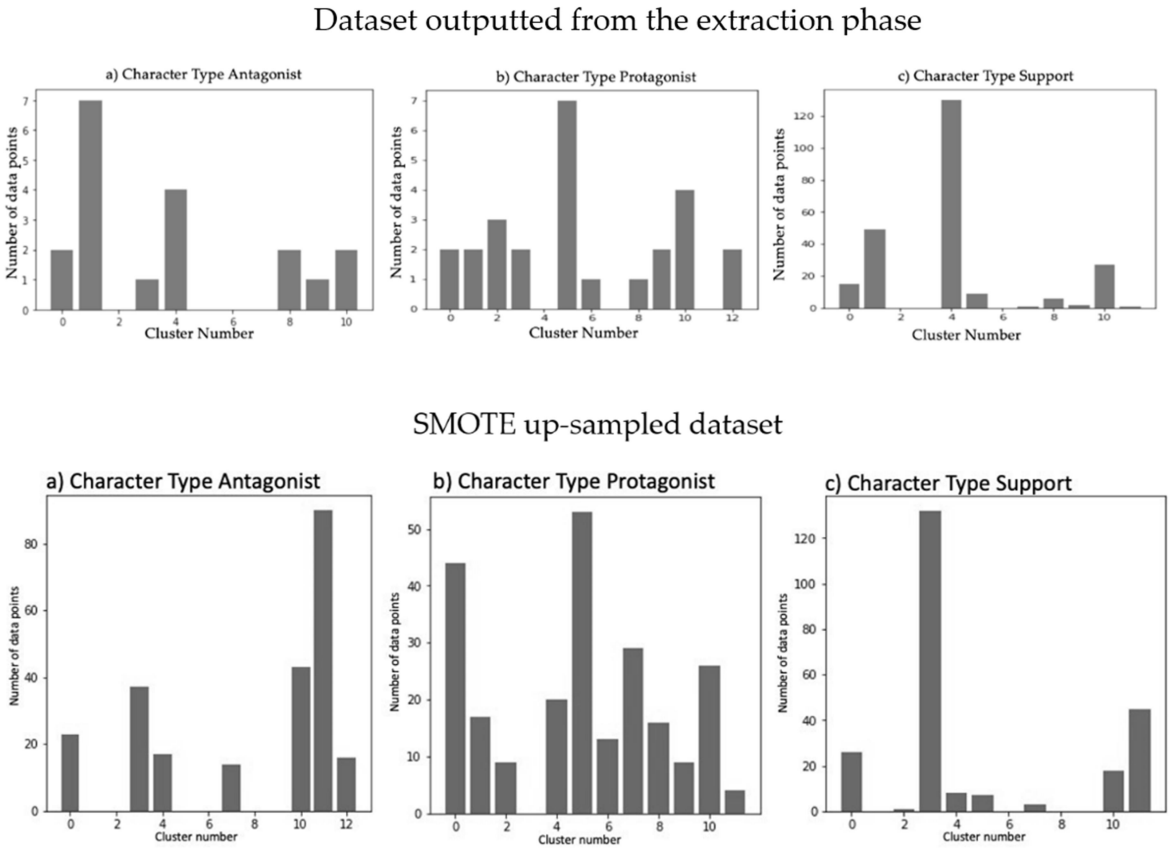

FIGURE 6: K-MEANS - CLUSTER ASSIGNMENTS OF DATA POINTS PER CHARACTER TYPE

In the case of K-Means, the analysis of the cluster formation on the dataset outputted from the extraction stage shows that the data points of type antagonist were assigned to seven clusters: the cluster with the most antagonists contained seven samples out of 19, which represents approximately 36.84% of the total number of antagonist characters, while two of the clusters contained only one sample each, representing approximately 5.27% each of all antagonists. When the dataset was upsampled, the distribution of antagonists was improved, despite that they were distributed across seven clusters again: the cluster with the most antagonists contained 102 samples out of 240 antagonists, representing 42.5% of all antagonists, while the cluster with the lowest number of antagonists contained 14 samples, accounting for 5.83% of all antagonists in the upsampled version. These results indicate that a larger dataset, as well as additional features, may be able to capture the similarities among antagonists more effectively.

K-Means allocated the protagonist characters to even more clusters, with 10 clusters in the dataset outputted from the extraction stage and 11 clusters in its upsampled version: again, the cluster with the most protagonists contained seven samples out of 26, which represents approximately 26.92% of the total number of protagonist characters, while two of the clusters contained only one sample each, representing approximately 3.85% each of all antagonists. Although there was an additional cluster found in the upsampled version of the dataset, the distribution was similar in that the biggest cluster accounted for 22.08% (53 samples) of all 240 support characters, while the smallest cluster accounted for 3.75% (nine samples) of the 240

support characters. Taken together, the results indicate that protagonists showed the greatest level of dispersion across both datasets, indicating that the features we derived failed to capture a great level of similarity across protagonists, irrespective of the number of samples.

With respect to the support characters, despite that we expected a greater level of diversity in this group, K-Means formed fewer clusters than for protagonists, that is, nine clusters on both datasets. Additionally, the characters cluster assignment appears to be more consistent across the two datasets, with one cluster accounting for more than half of the samples: the biggest cluster in both datasets contained 130 samples out of 240 support characters, accounting for 54.16%, while the smallest cluster contained only one character, representing approximately 0.42%. These results indicate that support characters showed the greatest level of similarity across the features we derived.

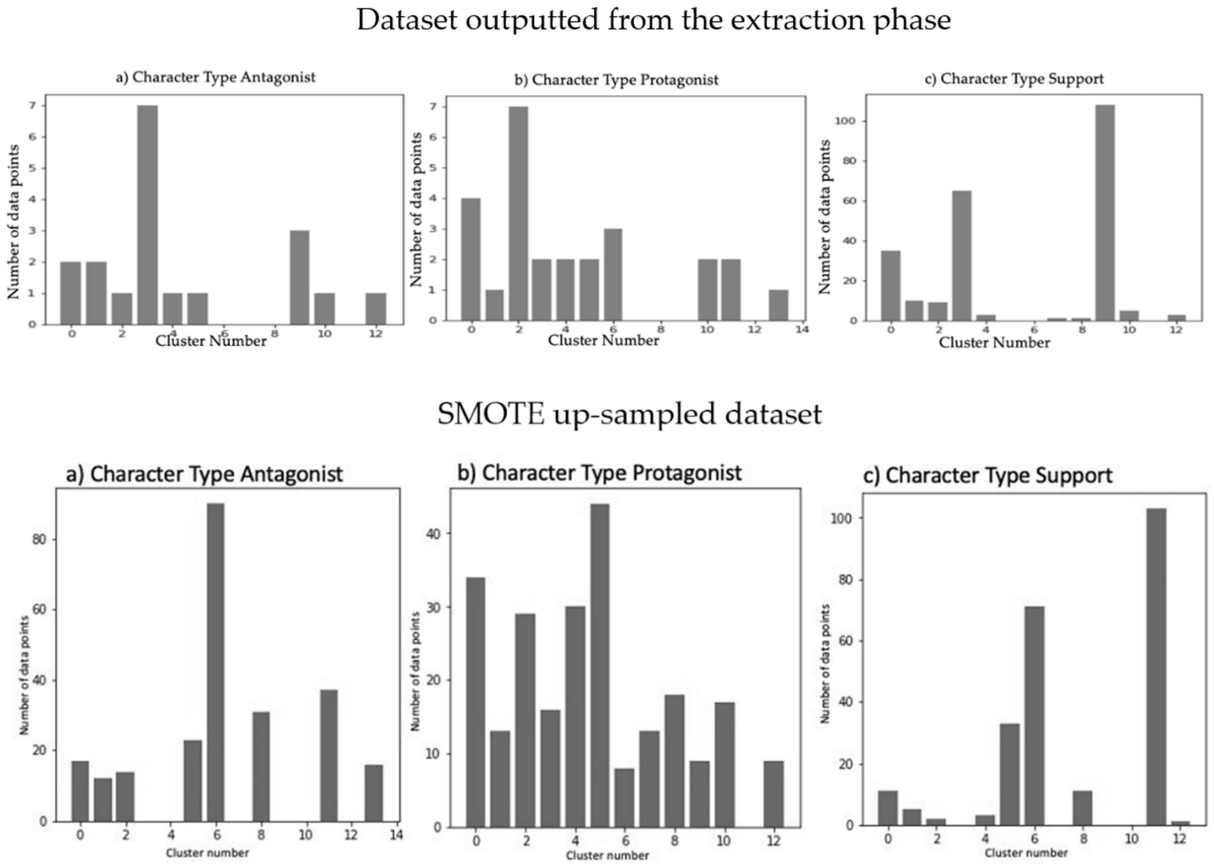

FIGURE 7: AGGLOMERATIVE CLUSTERING - CLUSTER ASSIGNMENTS OF DATA POINTS PER CHARACTER TYPE

Agglomerative Clustering consistently allocated characters to more clusters than K-Means, across all types: nine antagonist clusters on the dataset outputted from the extraction stage and eight antagonist clusters on the upsampled version, as opposed to seven antagonist clusters on both datasets; 10 protagonist clusters on the dataset outputted from the extraction stage and 12 protagonist clusters on the upsampled version, as opposed to 10 and 11 protagonist clusters, respectively, formed by K-Means; and, finally 10 support clusters on the dataset outputted from the extraction stage and nine support clusters on the upsampled version, as opposed to nine support clusters formed by K-Means on both datasets. Despite this, Agglomerative Clustering showed similar clustering patterns to K-Means, with support characters demonstrating the

most level of similarity across the features we derived, while protagonists displayed the least similarity. Moreover, in the case of antagonists, on the dataset outputted from the extraction stage, the cluster with the most and least antagonists contained the same number of samples as those from K-Means, seven antagonist samples and one antagonist sample, respectively.

To understand whether the clusters contain characters that are semantically related, we further investigated the level of overlap within the twelve K-Means clusters[2] generated on the dataset outputted from the extraction phase, as shown in Figure 7.

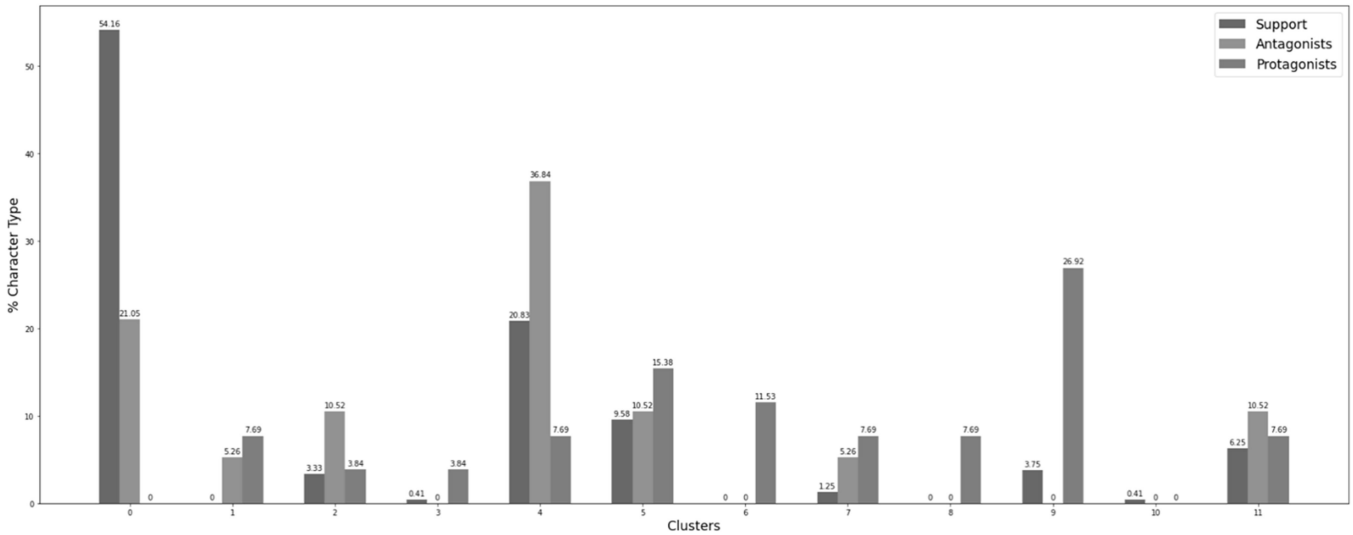

FIGURE 8: K-MEANS – PERCENTAGE OF EACH CHARACTER TYPE PER CLUSTER

As can be seen, the first cluster (cluster 0) contained the most characters being dominated by 130 support characters (approx. 54.16% of all support characters), but it also contained four antagonist samples (approx. 21.05% of all antagonists), with no protagonists being allocated to this cluster. The second biggest cluster (cluster 4) contained 59 samples, among which 50 samples represented support characters (approx. 20.83% of all support characters), seven samples represented antagonists (approx. 36.84% of all antagonists), and two samples represented protagonists (approx. 7.69% of all protagonists). These findings indicate that support and antagonist characters are more likely to be semantically related since they tend to appear together in a greater proportion. On the other hand, protagonists, despite that they are spread across more clusters, they tend to appear either in isolation (e.g., clusters 6 and 8, respectively), or together with small proportions of anatagonists (e.g., clusters 3 and 9, respectively), indicating that protagonist characters are less likely to be semantically related to each other and to the other types of characters. Half of the clusters contained all characters types – clusters 1, 2, 4, 5, 7, and 11, although each type represented by smaller proportions in each of these, indicating that approximately 41.24% of support characters, 73.66% of antagonists, and 42.29% of protagonists could not be identified as distinct types by the KMeans algorithm using the features we derived.

[2] Note that cluster numbering is arbitrary and the numbers assigned to each cluster do not carry any semantic information.

5. Conclusions

In this paper, we presented a novel heuristic-based approach to extracting characters from summarised story plots. As the results indicated, our approach successfully extracted characters with an average F1-score of 0.86, whereas the baseline approach based on basic CoreNLP only achieved an average F1-score of 0.63. Additionally, the results indicate an improvement in previously published work in the area of heuristic-based character extraction. Vala et al. (2015), who also used two of the summarised story plots from SparkNotes.com - 'Moonstone' and 'Pride and Prejudice', reported a recall value of 0.599 for both stories, while our approach had a recall value of 1 for both stories, showing that our approach successfully extracted all characters; moreover, although the precision values were not explicitly reported by Vala et al. (2015), they were said to be of lower value, whereas our approach achieved a precision of 0.875 for 'Moonstone' and a value of 0.963 for 'Pride and Prejudice'.

We have also successfully identified the types of characters – protagonist, antagonist, and support - applying a multi-class classification approach to supervised learning to a dataset we built across a novel set of 26 features. While the baseline approach only achieved an average F1-score of 0.39, with values as low as 0 in the case of protagonists for all algorithms, our approach achieved an average F1-score of 0.94., in many cases achieving above 0.95 F1-scores. However, the unsupervised approach failed to cluster these characters into three clusters, with an average V-measure score of only 0.277, despite that it constitutes an improvement from the unsupervised baseline approach which achieved a low average V-measure score of 0.127, indicating that other features are needed to capture similarities to a greater extent among characters of the same type, especially in the case of protagonists and antagonists, as demonstrated by the cluster analysis from the results of K-Means and Agglomerative Clustering. Based on the results we obtained through our approach, we can conclude that supervised learning is the most suited in identifying character types from the feature set we built using NLP and heuristic rules.

Nevertheless, our paper also identified several gaps that could be addressed by future research. In terms of character extraction, our error analysis found that our approach lead to 48 false positives and 51 duplicated PNEs. These were mainly due to the fact that animacy confirmation against WordNet database failed in many cases, to the fact that the last name resolution heuristic failed to identify aliases when a character was referred to by the last name only, and to the fact that gender resolution was inaccurate when a character was referred to using different honorifics, indicating that additional improvements in these areas may be targeted by future work.

In terms of character type identification, although the classification models performed well, future work can be extended to larger datasets, to evaluate additional grammatical, contextual, and statistical features, as well as to evaluate a finer-grained character set based on a different taxonomy, such as the one described by Talib (2010). Additionally, future research could explore the character type classification as a text classification task using traditional bag of

words models such as unigrams or bigrams with various weighting schemes (e.g., TF-IDF), as well as sequential models such as word embeddings and transformer architectures (Cholet 2021).

On the other hand, the feature set we developed did not enable the clustering algorithms to perform well and, as such, future work could focus on determining the optimal feature set by understanding whether additional features or different features play an important role in defining the clusters. Additionally, feature selection techniques may be investigated to understand whether a subset of the 26 features may lead to better cluster definition, as well as text-mining derived features such as TF-IDF (Chaturverdi et al. 2016) to understand whether text features could contribute to clustering performance. Finally, future research may focus on investigating various cluster formations from a literary and grammatical perspectives to understand how characters belonging to the same clusters are semantically related; for instance, what literary mechanisms and semantic features makes a clustering algorithm place characters such as *Turtle* (a support character from *The Bean Trees*), *Shawn* (an antagonist from *Educated*), and *Jonas* (a protagonist from *The Giver*) in the same cluster.

References

Agarwal, Apoorv, and Owen Rambow. 2010. "Automatic detection and classification of social events." In Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, 1024-1034.

Baccianella, Stefano, Andrea Esuli, and Fabrizio Sebastiani. 2010. "Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining." In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*.

Bamman, David, Brendan O'Connor, and Noah A. Smith. "Learning latent personas of film characters. 2013." In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics 1,* 352-361.

Calix, Ricardo A., Leili Javadpout, Mehdi Khazaeli, and Gerald M. Knapp. 2013."Automatic detection of nominal entities in speech for enriched content search." In *The Twenty-Sixth International FLAIRS Conference*.

Ceri, Stefano, Piero Fraternali, Aldo Bongio, Marco Brambilla, Sara Comai, and Maristella Matera. 2003. *Morgan Kaufmann series in data management systems: Designing data-intensive Web applications*. Morgan Kaufmann, 34-36.

Chaturvedi, Snigdha, Shashank Srivastava, Hal Daume III, and Chris Dyer. 2016. "Modelling evolving relationships between characters in literary novels." In *Proceedings of the AAAI Conference on Artificial Intelligence*, 30.

Chen, Yu-Hsin, and Jinho D. Choi. 2016. "Character identification on multiparty conversation: Identifying mentions of characters in tv shows." In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 90-100.

Chollet, Francois. 2021. *Deep Learning with Python*. New York: Manning Publication.

Feinerer, Ingo, Kurt Hornik, Mike Wallace, and Maintainer Kurt Hornik. 2020. "Package 'wordnet'."

Fernandez, Matt, Michael Peterson, and Ben Ulmer. 2015. "Extracting social network from literature to predict antagonist and protagonist." *Recuperado de: https://nlp. stanford. edu/courses/cs224n/2015/reports/14. pdf*.

Finkel, Jenny R., Trond Grenager, and Christopher Manning. 2005. "Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling". In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, 363-370.

Grandini, Margherita, Enrico Bagli, and Giorgio Visani. 2020. "Metrics for multi-class classification: an overview." *arXiv preprint arXiv:2008.05756*.

Hachey, Ben, Will Radford, Joel Nothman, Matthew Honnibal, and James R. Curran. 2013. "Evaluating entity linking with wikipedia." *Artificial intelligence* 194: 130-150.

Jahan, Labiba, Geeticka Chauhan, and Mark A. Finlayson. 2018. "A new approach to animacy detection." In *Proceedings of the 27th International Conference on Computational Linguistics*.

Jahan, Labiba, and Mark Finlayson. 2019. "Character identification refined: A proposal. " In *Proceedings of the First Workshop on Narrative Understanding*, 12-18.

Jahan, Labiba, Rahul Mittal, W. Victor Yarlott, and Mark Finlayson. 2020. "A straightforward approach to narratologically grounded character identification." In *Proceedings of the 28th International Conference on Computational Linguistics*, 6089-6100.

Jung, Jason J., Eunsoon You, and Seung-Bo Park. 2013. "Emotion-based character clustering for managing story-based contents: a cinemetric analysis." *Multimedia tools and applications* 65: 29-45.

Kong, Fang, Guodong Zhou, Longhua Qian, and Qiaoming Zhu. 2010. "Dependency-driven anaphoricity determination for coreference resolution." In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, 599-607.

Labatut, Vincent, and Xavier Bost. 2019. "Extraction and analysis of fictional character networks: A survey." *ACM Computing Surveys (CSUR)* 52: 1-40.

Liang, Tyne, and Dian-Song Wu. 2004. "Automatic pronominal anaphora resolution in English texts." In *International Journal of Computational Linguistics & Chinese Language Processing, Volume 9, Number 1, February 2004: Special Issue on Selected Papers from ROCLING XV*, 21-40.

Manning, Christopher D., Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. "The Stanford CoreNLP natural language processing toolkit." In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, 55-60.

Rosenberg, Andrew, and Julia Hirschberg. 2007. "V-measure: A conditional entropy-based external cluster evaluation measure." In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, 410-420.

Sukthanker, Rhea, Soujanya Poria, Erik Cambria, and Ramkumar Thirunavukarasu. 2020. "Anaphora and coreference resolution: A review." *Information Fusion* 59: 139-162.

Talib, Ismail S. 2010. "Narrative theory: A brief introduction." https://courses.nus.edu.sg/course/ellibst/NarrativeTheory/ .

Vala, Hardik, David Jurgens, Andrew Piper, and Derek Ruths. 2015. "Mr. bennet, his coachman, and the archbishop walk into a bar but only one of them gets recognized: On the difficulty of detecting characters in literary texts." In *Proceedings of the 2015 conference on empirical methods in natural language processing*, 769-774.

Valls-Vargas, Josep, Santiago Ontanón, and Jichen Zhu. 2014. "Toward automatic character identification in unannotated narrative text." In *Seventh intelligent narrative technologies workshop*.