Language Resources and Evaluation (2023) 57:849–867
https://doi.org/10.1007/s10579-022-09595-3

ORIGINAL PAPER

Detecting explicit lyrics: a case study in Italian music

Marco Rospocher[1]

Accepted: 27 April 2022 / Published online: 21 May 2022

Abstract
Preventing the reproduction of songs whose textual content is offensive or inappropriate for kids is an important issue in the music industry. In this paper, we investigate the problem of assessing whether music lyrics contain content unsuitable for children (a.k.a., explicit content). Previous works that have computationally tackled this problem have dealt with English or Korean songs, comparing the performance of various machine learning approaches. We investigate the automatic detection of explicit lyrics for Italian songs, complementing previous analyses performed on different languages. We assess the performance of many classifiers, including those–not fully exploited so far for this task–leveraging neural language models, i.e., rich language representations built from textual corpora in an unsupervised way, that can be fine-tuned on various natural language processing tasks, including text classification. For the comparison of the different systems, we exploit a novel dataset we contribute, consisting of approximately 34K songs, annotated with labels indicating explicit content. The evaluation shows that, on this dataset, most of the classifiers built on top of neural language models perform substantially better than non-neural approaches. We also provide further analyses, including: a qualitative assessment of the predictions produced by the classifiers, an assessment of the performance of the best performing classifier in a few-shot learning scenario, and the impact of dataset balancing.

Keywords Neural language models · Convolutional neural networks · Text classification · Explicit content detection · Italian language

1 Introduction

An important moral duty of our modern society is to prevent the exposure of young people to content (e.g., language, images, movies) that may be offensive or unsuitable for them, typically referred to as *explicit content*. When dealing with language,

✉ Marco Rospocher
marco.rospocher@univr.it

[1] Università degli studi di Verona, Lungadige Porta Vittoria, 41, 37129 Verona, Italy

such content includes: "strong language; references to violence, physical, or mental abuse; references to sexual behaviour; discriminatory language."[1] Even in the music business, over the years, organizations such as the Recording Industry Association of America (RIAA) have recommended the use of labels, such as PAL,[2] to mark that the content of a music product (mainly lyrics, but also booklets or related products) may be hurtful or inappropriate for children. Protecting kids from the exposure to such content is even more pressing nowadays with the widespread diffusion on the Web of easy-to-access digital content providers (e.g., Amazon Music, Spotify, YouTube) that deliver millions of songs. Indeed, many of these online platforms label with some dedicated tag (e.g.,"explict lyrics" or "E") songs whose reproduction to kids should be prevented. However, this labelling mainly results from human work (from "record labels, industry partners and our users"[3]) and platforms explicitly declare the unfeasibility to tag all (actual) explicit content.[4]

Recently, some works have proposed to tackle the problem of identifying lyrics containing offensive content by means of computational methods, either exploiting dictionary based approaches or machine learning (ML) classifiers, including neural approaches. More in details, Chin et al. (2018) and Kim and Yi (2019) have studied the problem working on Korean lyrics, while other works have compared the performance of various systems on English (Bergelid, 2018; Fell et al., 2019; Rospocher 2021), although none have dealt so far with Italian songs.

The contribution of this work is manifold. First, we contribute the first Italian language dataset for explicit song lyrics detection. Second, we empirically assess the performance of several classification methods on the given dataset, from classical ones to recent approaches leveraging neural language models, the latter being only preliminary tested–BERT (Devlin et al., 2019) for English lyrics in Fell et al. (2019)–with unsatisfactory performance. We complement this comparative performance analysis with a detailed assessment of the quality of the prediction provided by the classifiers. Third, given the best performing approach, we further analyse its capability to perform well with a limited amount of training samples, an experimental setting known as few-shot learning, and when trained with different proportions of explicit / non-explicit content. Fourth, all the developed models, including the fine-tuned neural language models, are made available so that they can be freely used and applied for detecting unseen explicit lyrics.

The paper is organized as follows. Section 2 overviews the relevant related work on detecting explicit content in music lyrics. Section 3 briefly presents all the methods considered in this paper, including the neural language models for Italian and their fine-tuning for text classification. Section 4 presents the conducted performance assessment, describing the novel contributed dataset, the evaluation methodology, and the performance results. Section 5 discusses some of the findings of

[1] https://support-en.imusician.pro/article/154-what-is-considered-as-explicit-content.
[2] https://www.riaa.com/resources-learning/parental-advisory-label/.
[3] c.f. Deezer: https://support.deezer.com/hc/en-gb/articles/360000590898-Explicit-content.
[4] c.f. Spotify: https://support.spotify.com/us/article/explicit-content/.

the evaluation and additional assessments on the quality of the predictions, few-shot learning, and applying dataset balancing, while Sect. 6 concludes.

2 Related work

In the last four years, Natural Language Processing (NLP) techniques have been effectively applied in different works to detect offensive content in music lyrics, to prevent the reproduction of hurtful or inappropriate songs for children.

Two previous works have studied the problem on Korean lyrics. First, Chin et al. (2018) compared various approaches on a dataset of 27,695 Korean song lyrics (3.7% of the which marked as explicit): from simple techniques such as spotting words from a profanity language dictionary to more advanced methods based on machine learning (ML) classification algorithms (AdaBoost et al., 2006). Bagging scored best in the conducted evaluation ($F_1 = 0.78$). Later on, Kim and Yi (2019) addressed the problem on a corpus of 70,077 Korean song lyrics (10.7% of the which marked as explicit). Besides experimenting with a lexicon-based filtering strategy, leveraging an automatically built dictionary of explicit words, the authors also exploited Hierarchical Attention Networks (HAN) (Yang et al., 2016), an RNN-based model for sequential and hierarchical processing of words, showing that the latter, when combined with vector representations modelling the occurrence of words from the automatically generated lexicon of explicit terminology, scores the best performance ($F_1 = 0.805$).

Recent works have also studied the problem on English lyrics. Bergelid (2018) compared the performance of several classical ML algorithms–linear Support Vector Machine (LSVM) (Cortes & Vapnik, 1995), Multinomial Naive Bayes (MNB) (Kibriya et al., 2004), k-Nearest Neighbors (KNN) (Fix et al., 1951), Random Forest (RF) (Rokach, 2010)—on a corpus consisting of 25,441 English song lyrics (13% of the which marked as explicit). Lyrics were initially preprocessed with vectorization techniques –TF-IDF (Sparck Jones, 1988), Doc2Vec (Le & Mikolov, 2014)—in order to extract the features for the classifiers. Highest scores were achieved with LSVM and MNB, starting from TF-IDF vectors ($F_1 = 0.677$). A further performance improvement ($F_1 = 0.826$) was achieved with RF by balancing the dataset via undersampling techniques (i.e., randomly removing non-explicit lyrics from the dataset in order to have the same amount of explicit and non-explicit lyrics). The authors also observed a marginal improvement by extending the features fed to the classifiers with content beyond lyrics, such as the artist name or the music energy level.

Fell et al. (2019) compared various machine and deep learning algorithms for explicit lyrics classification on a new dataset of 179,391 English song lyrics (9.9% of the which marked as explicit): a logistic regression classifier fed with TF-IDF BOW vector representations, BERT Language Model (Devlin et al., 2019), and Textual Deconvolution Saliency (Vanni et al., 2018), a CNN for text classification. The work shows that deep models do not outperform the other shallow approaches, and logistic regression ($F_1 = 0.780$), performing even slightly better than BERT

($F_1 = 0.777$), is considered a strong baseline in terms of performance and computational costs.

Finally, Rospocher (2021) assessed the performance of the fastText word embeddings and classifier, on the largest dataset considered so far, consisting of 807,707 English song lyrics (7.74% of the which marked as explicit). The work shows that fastText outperforms several baselines (e.g, majority vote, logistic regression), suggesting that further improvement may be achieved by exploiting the fastText representations with more advanced classifiers (e.g., 1D-CNN).

In our work we further extend the state-of-the-art by studying the problem of detecting offensive music lyrics written in Italian. We contribute a new dataset consisting of Italian songs and assess the performance of several classifiers on it, including various recent neural language models, not yet fully exploited for this task.

3 Text classifiers based on machine learning

The goal of the work is to assess the capabilities of state-of-the-art techniques, including neural language models, in automatically detecting explicit Italian song lyrics, a problem that can be formulated as a binary classification task in a supervised setting, where available annotated data (i.e., lyrics tagged with explicitness information) is used to train a system able to predict unseen song lyrics to one of two classes (explicit vs. non-explicit).

Various techniques proved effective over the years to tackle binary classification tasks when applied to written text. Also in light of previous works on detecting explicit song lyrics, in our assessment we consider the following approaches.

3.1 Logistic Regression

Logistic Regression (LR) is a Machine Learning method that exploits a discriminative classifier based on linear equations to assign input data to candidate classes. It is a traditional baseline as supervised machine learning algorithm for text classification, that demonstrated strong, competitive performance for detecting explicit English lyrics (Fell et al., 2019). Lyrics have to be converted into numerical features, as LR expects fixed-size numerical vectors in input. A common strategy is to apply a bag-of-words (BOW) vectorization technique such as TF-IDF (Sparck Jones, 1988).

3.2 fastText classifier

Facebook AI released fastText (Bojanowski et al., 2017), a library for efficient learning of word representations. Differently from other word representation learning approaches, fastText associates to each word a bag of character *n*-grams (a.k.a., *subwords*), and the representation of the word results from the sum of the vector representations of its *n*-grams. This way, fastText can compute word representations also for words never seen during the learning phase of the representation (by combining subwords), enabling to better deal with languages with large vocabularies and

rare words, in particular those unseen during the training phase. Building on this word representation learning approach, an efficient document classifier was later proposed (Joulin et al., 2017): it adopts multinomial logistic regression, where each input document is encoded as a document vector, obtained by averaging the FASTTEXT word representations of all the words in it. The FASTTEXT classifier was evaluated for explicit song lyrics detection in Rospocher (2021), showing that it outperforms the LR classifier fed with the TF-IDF vectors on English lyrics.

3.3 1D-CNN

Originally foreseen for 2D data such as images and videos, Convolutional Neural Networks (CNN) (LeCun et al., 1998) are deep learning architectures that exploit convolution for capturing spatial and temporal dependencies in the input data. However, as observed in Kim (2014), CNNs can be adapted (the so-called 1D-CNN) also to 1-dimension data such a text, and proved effective for text classification (e.g., question classification, sentiment analysis). Similarly to LR, a 1D-CNN can be fed with word vectors obtained from an unsupervised (neural) language model. Similarly to what done in Rospocher (2021), we fed the 1D-CNN with FASTTEXT vectors, a configuration that showed promising results on English lyrics.

3.4 Classifiers built on top of Neural Language Models

Neural Language Models (NLMs) are large-scale pre-trained deep language representation models that can be exploited to perform many natural language understanding tasks. Their development is enabled by Transformers (Vaswani et al., 2017), neural Seq2Seq architectures that leverage attention to capture dependencies between input and output sequences. Once pre-trained (on a generic task such as masked-language modelling), a NLM can be fine-tuned to the specific NLP task considered: this boils down to adapt and train the last output layer of the transformer network, thus retaining (and enabling to exploit) for the task all the semantics and knowledge of the language captured in the previous layers of the network

Most prominent examples of NLMs include BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), mainly developed for English. A first experiment on using BERT for explicit lyrics classification, in English language, was conducted by Fell et al. (2019), showing performance slightly lower than other considered approaches. However, as in this work we deal with Italian text, we consider and compare the following NLMs suitable for Italian:

– Multilingual-BERT (Devlin et al., 2019):[5] developed by Google, leverages a deep bidirectional representation learned from unlabelled text from Wikipedia in multiple languages. It is a single language model pre-trained from monolingual corpora in 104 languages, including Italian;

[5] https://github.com/google-research/bert

- Italian-BERT (Schweter, 2020):[6] developed by the MDZ Digital Library team (dbmdz) at the Bavarian State Library, is a BERT model for Italian trained on Italian Wikipedia and various texts from the OPUS corpora collection;
- AlBERTo (Polignano et al., 2019):[7] developed by the University of Bari, is a BERT language model for Italian, trained on the Italian language used in social media (Twitter);
- UmBERTo-Commoncrawl and UmBERTo-Wikipedia:[8] developed by Musixmatch AI, are two RoBERTa-based Language Models trained on different Italian corpora. The first is a cased model trained on Commoncrawl ITA exploiting OSCAR (Open Super-large Crawled ALMAnaCH coRpus), while the second is an uncased model trained on a corpus extracted from the Italian Wikipedia;
- GilBERTo:[9] is an Italian pre-trained RoBERTa-based language model, leveraging the CamemBERT text tokenization approach. It is trained on OSCAR.

In order to apply NLMs for detecting explicit lyrics, first lyrics have to be appropriately tokenized. A sigmoid-activated sequence classification layer is then added on top of each pre-trained model and fine-tuned for the task. All models are exploited through the HuggingFace[10] Python library interface.

4 Evaluation

The goal of the work is to assess the performance of different classifiers, including recent ones built on NLMs, in detecting explicit Italian music lyrics. All the materials–dataset, Python code for all the methods, trained model of each system, predicted labels on the test set, scores–are publicly released at https://github.com/rospocher/explicit-lyrics-detection, both to make all the experiments reproducible but also to provide ready-to-use tools to be used for predicting unseen explicit lyrics.

4.1 Dataset of Italian lyrics annotated with explicitness

To train and assess the various approaches presented in Sect. 3, a dataset consisting of song lyrics annotated with explicitness information (i.e., whether the lyrics contain explicit content or not) is needed. To the best of our knowledge, no such dataset is available and distributed for Italian song lyrics. To build such dataset we rely on content provided through public platforms, namely LyricWiki and Spotify.

[6] https://github.com/stefan-it/italian-bertelectra.
[7] https://github.com/marcopoli/AlBERTo-it.
[8] https://github.com/musixmatchresearch/umberto.
[9] https://github.com/idb-ita/GilBERTo.
[10] https://huggingface.co/.

More precisely, we start from a recent dump of LyricWiki [11] made available through Internet Archive.[12]

We process the XML dump to select only the pages containing song lyrics (i.e., having the tag <lyrics>), thus excluding pages about artists or albums. From these pages, we extract the content of the <lyrics> element, together with the metadata information about the song (from the header and footer template of the page), such as artist, title, year, language, and the links to other resources, such as iTunes, Spotify, YouTube, etc.. We automatically pre-process the text of the lyrics removing possible tags (e.g., HTML code) or annotations (e.g., wiki markup denoting song sections), and dropping songs with empty lyrics (e.g., among them, instrumental tracks). Furthermore, of all the resulting song lyrics, we keep only the ones tagged as Italian language and having a corresponding Spotify ID. As we spotted some songs tagged as Italian but containing non-Italian text, we further run the lyrics through a language prediction tool,[13] keeping only those predicted as Italian.

Exploiting the Spotify API, we retrieve the metadata information for the track from Spotify, including the "explicit" boolean value (0=non-explict; 1=explicit), thus obtaining a dataset of lyrics text with the corresponding explicitness information. The characteristics of the resulting dataset are reported in Table 1.

As expected, the dataset is unbalanced, as non explicit lyrics are many more than explicit ones: note that the percentage of explicit lyrics in the dataset is 6.04, a value consistent with the ones of the datasets for other languages considered in the literature (c.f. Sect. 2). We randomly split the dataset in a training and testing part following the standard Pareto 80-20 proportion, preserving the ratio between explicit and non-explicit lyrics. The stats of the two splits are reported in Table 1. The train and test splits are made available in the evaluation material.[14]

4.2 Research questions and evaluation measures

The research questions we aim to address are the following:

RQ1 What is the best performing classifier for detecting explicit Italian song lyrics?
RQ2 Do the NLM classifiers outperform other competitive methods for explicit Italian lyrics detection? Does one of the assessed NLM classifiers perform substantially better than the others for the considered task?

[11] LyricWiki was shut-down in late 2020 (https://web.archive.org/web/20200830142257/https://lyrics.fandom.com/wiki/LyricWiki.), after the data was collected, but its content is still available through Internet Archive.

[12] https://archive.org/download/wiki-lyricsfandomcom/lyricsfandomcom-20200216-history.xml.7z.

[13] whatthelang: https://github.com/indix/whatthelang.

[14] Due to licensing issues, besides the explicitness metadata information, we can only make available the LyricWiki page ID of each lyrics, from which the full text of the lyrics can be retrieved from the Internet Archive dump.

Table 1 Details on the size and splits of the evaluation dataset

Sources		LyricWiki, Spotify	
Year coverage		1950-2019	
Language		Italian	
% Explicit lyrics		6.04	
	Non-explicit	Explicit	Total
Train	25,286	1622	26,908
Test	6321	406	6727
	31,607	2028	33,635

To address these research questions, we compute the standard classification metrics: precision P, recall R, and F_1 score. We first compute these metrics separately on each class i (1 identifies the explicit class / 0 = identifies the non-explicit class), denoting them with P^i, R^i and F_1^i: P^i measures the precision of the classifier on the class (how many of the predicted lyrics in that class are correct); R^i measures how well the classifier covers the class (how many of the lyrics to be assigned to that class are predicted as such by the classifier); and, F_1^i returns a single representative value combining the two.

To aggregate the performance on the two classes we macro-average the corresponding measures (i.e., $P = \frac{P^0+P^1}{2}$, $R = \frac{R^0+R^1}{2}$, $F_1 = \frac{F_1^0+F_1^1}{2}$). For completeness, we also report accuracy (A : i.e., the percentage of correctly predicted classes of the test set) and weight-averaged metrics (wP , wR , wF_1 : i.e., metrics scaled according to the relative number of samples in the classes), but given the unbalanced nature of the dataset, macro-averaged metrics better capture the performance on *both* classes, while the others are more biased toward the most represented class. All the metrics are computed using the Python scikit-learn classification_report method.[15]

4.3 Compared systems and evaluation procedure

We compute and compare the performance of several classifiers on the given dataset, those introduced in Sect. 3 as well as few additional baselines:

- Majority: a system that always predicts the majority class (i.e., non-explicit) for any given lyrics;
- Dictionary: a system that predicts explicit lyrics if they contain words from a dictionary of offensive Italian terms. For the dictionary, we use one already compiled containing 500 badwords;[16]
- LR-BOW: a logistic regression classifier, where the word features are weighted with the well-known TF-IDF scheme;

[15] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html.
[16] https://github.com/napolux/paroleitaliane/blob/master/paroleitaliane/lista_badwords.txt.

- FASTTEXT: the FASTTEXT classifier, leveraging pre-trained word embeddings for Italian.[17] We also experimented with learning the embeddings from the lyrics text as done in Rospocher (2021), but the performance were substantially lower than with pre-trained embeddings, probably due to the limited amout of text for learning the representations;
- 1D-CNN$_{FT}$: the 1D-CNN classifier, fed with pre-trained FASTTEXT vector representations;
- MBERT$_C$: our fine-tuning of mBERT for lyrics classification;
- ITALIAN-BERT$_C$: our fine-tuning of Italian-BERT for lyrics classification;
- ALBERTO$_C$: our fine-tuning of AlBERTo for lyrics classification;
- UMBERTO-W$_C$: our fine-tuning of UmBERTo-Wikipedia for lyrics classification;
- UMBERTO-CC$_C$: our fine-tuning of UmBERTo-Commoncrwal for lyrics classification;
- GILBERTO$_C$: our fine-tuning of GilBERTo for lyrics classification.

All systems are tested on the same samples (test split) and trained (except those not requiring training: Majority and Dictionary) on the same samples (train split of the dataset), using a 20% portion of the training part for optimizing the hyper-parameters of the classifiers, and choosing the best model for each NLM classifier.

We report here the final configurations used, including the tuned hyper-parameters, for assessing the performance of the various systems (we recall that the Python code of each system is provided in the evaluation material):

- LR-BOW: for building the TF-IDF BOW vectors, we used the Python scikit-learn TfidfVectorizer class[18] with the default values except for the following parameters: *ngram_range* $= (1, 2)$ (i.e., both unigrams and bigrams are extracted),[19] *max_features* $= 100000$, *max_df* $= 0.95$, *min_df* $= 2$. For the logistic regression model, we used the Python scikit-learn LogisticRegression class[20] with the default values except for the following parameters: *max_iter* $= 2000$, *class_weight* $=$ *balanced*, *random_state* $= 42$;
- FASTTEXT: we used the Python fasttext library[21] with the default values for the classifier except for the following parameters: *lr* $= 0.042$, *epoch* $= 40$, *dim* $= 300$, *loss* $=$ *hs*, *wordNgrams* $= 4$, *minn* $= 4$, *maxn* $= 6$, *minCount* $= 1$;
- 1D-CNN$_{FT}$: we used the Python keras library.[22] The embedding layer is build with the following configuration: *num_word* $= 40,000$, *max_sequence_length* $= 500$, *epoch* $= 5$, *batch_size* $= 128$. The CNN includes 3 convolution layers (each set with default values except for: *filters* $= 128$, *kernel_size* $= 5$, *activation* $=$ *relu*), intertwined with max pooling layers (default

[17] https://fasttext.cc/docs/en/crawl-vectors.html.
[18] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html.
[19] Unigram-only and bigram-only variants were also tested, yielding comparable performance.
[20] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html.
[21] https://fasttext.cc/.
[22] https://keras.io/.

values except for: *pool_size* = 5), and a final global max pooling layer (default values) before the softmax activation layer;
– NLM classifiers: we used the Python HuggingFace library[23] to build all the NLM classifiers: they are obtained adding a sigmoid activated classification layer on top of each of the considered NLMs, and fine-tuned with a learning rate of 2×10^{-5} and with batches of 16 sequences, each having a maximum length of 512.

4.4 Results

Table 2 reports the performance of the various systems considered for assessing the research questions.

For non-NLM classifiers (first half of the table), the best scoring system is 1D-CNN$_{FT}$, followed by LR-BOW ($F_1 = -0.010$), Dictionary ($F_1 = -0.035$), fastText ($F_1 = -0.073$). They all perform substantially better than the trivial Majority baseline. The CNN model confirms to be a strong competitor for the task, as observed also in Fell et al. (2019) and Rospocher (2021) for English lyrics. Quite surprisingly, the scores of fastText are substantially lower than all other baselines (except Majority): for English lyrics, fastText proved very effective in Rospocher (2021), outperforming LR-BOW and performing on par with 1D-CNN$_{FT}$. This different behaviour may be due to the different size of the datasets used in the evaluation for the two languages ($\sim$ 34K vs $\sim$ 808K lyrics), which allowed training the fastText word representations directly on the English lyrics, while for Italian we had to resort to the pre-trained embeddings to achieve the best performance.

For NLM classifiers (second half of the table), the best scoring system is UmBERTo-CC$_C$, which perform substantially on par with AlBERTo$_C$ ($F_1 = -0.001$) and Italian-BERT$_C$ ($F_1 = -0.003$), with the other approaches achieving lower scores: GilBERTo$_C$ ($F_1 = -0.017$), UmBERTo-W$_C$ ($F_1 = -0.026$), and mBERT$_C$ ($F_1 = -0.108$). In particular, the scores of mBERT$_C$ are quite low with respect to the other NLMs: we recall that mBERT$_C$ is a multilingual model (differently from the others which are for Italian only) and maybe the size of the material used for fine-tuning the model is not enough for the task and language considered. Quite interestingly, the three best NLMs are pre-trained with substantially different corpora: OSCAR (UmBERTo-CC$_C$), Twitter (AlBERTo$_C$), and Wikipedia+OPUS (Italian-BERT$_C$).

Comparing the performance of non-NLM and NLM classifiers, we can see that all NLM classifiers (with the exception of mBERT$_C$) outperform the best scoring non-NLM classifiers (1D-CNN$_{FT}$). In particular, the best scoring NLM classifier (UmBERTo-CC$_C$) substantially outperform 1D-CNN$_{FT}$ ($F_1 = -0.041$), a difference which is statistically significant according to the McNemar's test (p-value < 0.01) (McNemar 1947). This somehow controverts the results on English lyrics reported in Fell et al. (2019), where BERT performed worse than shallow classifiers such as

[23] https://huggingface.co/.

Table 2 Classification scores of the tested systems, organized in two groups in the table: non NLM (first) and NLM (second) classifiers

System	Non-explicit			Explicit			Macro			Weighted			*A*
	P^0	R^0	F_1^0	P^1	R^1	F_1^1	P	R	F_1	wP	wR	wF_1	
Majority	.940	1.000	.969	.000	.000	.000	.470	.500	.484	.883	.940	.910	.940
Dictionary	.980	.939	.959	.424	.695	.527	.702	.817	.743	.946	.925	.933	.925
LR-BOW	.979	.953	.966	.486	.687	.569	.733	.820	.768	.950	.937	.942	.937
FASTTEXT	.957	.993	.975	.728	.310	.435	.843	.651	.705	.943	.955	.945	.951
1D-CNN $_{FT}$	.968	.987	**.977**	.708	.490	**.579**	.838	.739	**.778**	.952	.957	**.953**	**.957**
MBERT$_C$	.960	.983	.972	.582	.367	.450	.771	.675	.711	.937	.946	.940	.946
ITALIAN-BERT$_C$	.974	.986	.980	.728	.586	.649	.851	.786	.815	.959	.962	.960	.962
ALBERTO$_C$	.972	.991	**.982**	.806	.552	.655	.889	.772	.818	.962	.965	**.962**	**.965**
UMBERTO-W$_C$	.974	.978	.976	.630	.591	.610	.802	.784	.793	.953	.954	.954	.954
UMBERTO-CC$_C$	.975	.984	.980	.713	.611	**.658**	.844	.798	**.819**	.959	.962	.960	.962
GILBERTO$_C$	.974	.980	.977	.659	.599	.627	.816	.789	.802	.955	.957	.956	.957

Highest F_1/*A* scores in each group are marked in bold. For the Majority system, being P^1 and F_1^1 not computable, they are set to .000 by default by the scorer

LR-BOW and TDS (a CNN model), thus suggesting that findings of system evaluation on a language, at least for this task, do not necessarily hold for others.

Summing up, to answer our research questions, we can conclude from the results that UMBERTO-CC$_C$, ALBERTO$_C$, ITALIAN-BERT$_C$ are the best performing systems (RQ1) for the task on the considered dataset. Moreover (RQ2), Italian-specific NLMs clearly outperform all other approaches, with the UMBERTO-CC$_C$, ALBERTO$_C$, and ITALIAN-BERT$_C$ achieving the highest scores among them.

5 Discussion and additional findings

We complement the evaluation conducted to address our research questions with a few other analyses and considerations, useful for better assessing the classification performance for the considered task.

5.1 Computational aspects

From the computational point of view, the considered systems have different requirements. For fine-tuning (as well as inferencing with) NLMs, powerful processing units (e.g., GPU, TPU) are needed, while all other baselines can be easily trained and evaluated also on commodity hardware. Time performance wise, fine-tuning one of the NLMs on an online cloud machine powered by an Nvidia T4 GPU takes on average 165 minutes (inferencing on the test set takes

approximately 2 minutes), while training 1D-CNN$_{FT}$ on the same machine takes 2 minutes (inference on the test set takes approximately 25 seconds).

5.2 Qualitative analysis of the prediction performance

We had a closer look at the predictions produced by the systems involved in the evaluation focusing in particular on: UmBERTo-CC$_C$, the best scoring system according to the results reported in Sect. 4; 1D-CNN$_{FT}$, the best scoring non-NLM system; and LR-BOW, a reference system for explicit lyrics detection according to the related work.

All considered systems perform consistently well on the non-explicit class, with F_1^0 ranging from .966 (LR-BOW) to .980 (UmBERTo-CC$_C$). Besides some varying differences, the F_1^0 value for all systems results from substantially balanced values of P^0 and R^0, that is, substantially low values of false positives (i.e., lyrics predicted non-explicit but that are explicit) and false negatives (i.e., lyrics predicted explicit but that are non-explicit) for the non-explicit class, compared to the correctly predicted non-explicit lyrics.

Substantial differences can be observed instead on the explicit class, with F_1^1 ranging from .569 (LR-BOW) to .658 (UmBERTo-CC$_C$). First of all, we note that F_1^1 values are remarkably lower than the corresponding F_1^0, for each system. This is somehow unsurprising on a very unbalanced binary classification dataset like the considered one: as the false positives for one class become the false negatives for the other and viceversa, the main difference between the measures computed on one class and the other is the number of correctly predicted element of the class, which is inevitably much lower for the less represented one (i.e., the explicit class). Furthermore, in such an unbalanced scenario, ML systems typically tend to be biased toward the most represented class in the training data: balancing techniques may partially mitigate this phenomenon (c.f., "Balancing the training dataset" later in this section). Similar differences of performance on the two classes in detecting explicit lyrics are observed also for English language (Rospocher 2021).

We thus analysed in deep the predictions for explicit lyrics. There are 406 explicit lyrics in the test dataset:

- LR-BOW predicted 574 explicit lyrics (279 correct; 295 wrong), failing to predict other 127 explicit lyrics;
- 1D-CNN$_{FT}$ predicted 281 explicit lyrics (199 correct; 82 wrong), failing to predict other 207 explicit lyrics;
- UmBERTo-CC$_C$ predicted 348 explicit lyrics (248 correct; 100 wrong), failing to predict other 158 explicit lyrics.

That is: LR-BOW predicted much more explicit lyrics than the other two systems (more than double of 1D-CNN$_{FT}$), but more than half of them are actually non-explicit lyrics; 1D-CNN$_{FT}$ under-predicted explicit lyrics, missing more than half

of them; finally, UmBERTo-CC$_C$ predicted a number of explicit lyrics closer to the expected one than the other two systems, a high proportion of which are correct predictions, and missing a number of explicit lyrics halfway between the ones missed by the other two systems.

Of the 406 explicit lyrics, 100 were not predicted by any of the systems, while 178 were correctly predicted by all of them. Five of the latter where correctly predicted although not containing offensive Italian words (indeed, they were missed by the Dictionary baseline), meaning that, to some extent, all the three systems were somehow able to grasp a little bit the explicitness of the content beyond the usage of offensive words, but looking more at the context in which words are used. One example is the song "Killer Game"[24] by Salmo, which contains potential reference to use/abuse of drugs (e.g., smoking until getting hospitalized in a clinic) or dangerous practices (e.g., jumping off a balcony without safety measures, jumping down a precipice without a rope). Combining this with the numbers on explicit lyrics prediction previously reported, it looks like LR-BOW pushes a little bit more this generalization attempt to grasp the explicitness of the content according to the context (probably too much, looking at the number of wrong explicit predictions), 1D-CNN$_{FT}$ is much more conservative on this, making less errors but also covering less explicit songs, while UmBERTo-CC$_C$ stands in an intermediate, more-balanced position, better characterizing the explicit class. For instance, UmBERTo-CC$_C$ is the only system that correctly predicted the explicitness of the song "Polizia Una Razza da Estinguere"[25] by Cripple Bastards, that takes a strong, discriminatory position toward police, addressing it as a race, and claiming for its (physical) elimination, although none of the words of the song, taken individually, can be considered offensive.

Of the 100 explicit lyrics that were not predicted by any of the three systems, some of them (21) contains explicit words (captured for instance by the Dictionary baseline) while the remaining 79 are most probably marked explicit because of the way non-offensive words (per se) are actually used in an offensive context. Two examples of this kind are: the song "Nitroglicerina"[26] by Derozer, which contains reference of appreciation for abusing of dangerous substances (encouraging nitroglycerin intake by drinking); another example is "Il vicino"[27] by Punkreas, which contains reference to violence and hate (the neighbour is an enemy that has to be eliminated; dreaming to become a gas station attendant to sprinkle the neighbour with gasoline). All the words in these songs are not offensive per se (e.g., enemy, sprinkle, eliminate, nitroglycerin) and can be easily found also in non-explicit lyrics. This remarks once again that the detection of explicit lyrics goes beyond just finding the usage of specific words in a song, and understanding the context in which words are used is necessary for building effective systems for detecting lyrics unsuitable for children.

[24] https://testicanzoni.rockol.it/testi/salmo-feat-gemitaiz-madman-killer-game-115962397.
[25] https://testicanzoni.rockol.it/testi/cripple-bastards-polizia-una-razza-da-estinguere-13349737.
[26] https://testicanzoni.rockol.it/testi/derozer-nitroglicerina-61169430.
[27] https://testicanzoni.rockol.it/testi/punkreas-il-vicino-76511707.

Concerning the lyrics wrongly predicted as explicit by the systems (295 for LR-BOW; 82 for 1D-CNN$_{FT}$; 100 for UmBERTo-CC$_C$) we noticed that 35 of them where predicted the same by all the three systems (and also the Dictionary baseline). By a closer look at them, we feel that they indeed contain content unsuitable for children, although they are not marked as such in Spotify. Examples include: "La Pietra dello Scandalo"[28] by Primo & Squarta (with several references to sex, violence, offensive words) and "La colpa"[29] by Vacca (with references to women in a denigratory way, and sex). Indeed, as discussed in Sect. 1, it is acknowledged by Spotify that, as the information comes from right-holders, the platform cannot guarantee that all explicit content is actually tagged as such, so missing explicit labels in the dataset are expected (i.e., songs that are tagged non-explicit but actually contain explicit content). In our work, we decided to fully rely on the information already available in the considered platforms (LyricWiki, Spotify) without performing any manual annotation or revision. Clearly, a manual revision may indeed improve the quality of the dataset and, as a consequence, the performance of the classifiers. As a preliminary experiment in this direction, we simply re-run the evaluation of the three systems here considered fixing the annotation of the aforementioned 35 explicit lyrics. The results are reported in Table 3. As expected, the classification scores of all the systems improve, proportionally with the data already reported in Table 2, with UmBERTo-CC$_C$ scoring $F_1 = .850$.

Indeed, this preliminary experiment suggests a future work to improve the quality of the dataset with a post-editing, system-driven, strategy: i.e., to iteratively annotate (in a cross-fold manner) the dataset with all the considered systems in Sect. 4.3, and focus the manual revision (if needed) on the cases where many systems (e.g., two out of three, or three out of four) predict a label different than the one provided by Spotify.

5.2.1 Few-shot learning

NLMs are known for their capability to achieve high scores also when fine-tuned in situations of scarcity of task-specific annotated data, a setting typically known as *few-shot learning* (Brown et al., 2020). We decided to inspect if this also holds for the task of detecting explicit Italian lyrics, at least for the best performing NLM. We thus: (i) created various subsets (preserving the explicit/non-explicit lyrics ratio) of the train split of the dataset, of size in the range [1000, 25,000]; (ii) trained UmBERTo-CC$_C$ on each of these subsets; and, (iii) tested the resulting models on the whole test set of the dataset. A plot showing the trend of the F_1 when varying the size of the dataset is shown in Fig. 1.

The plots shows that at least 20K samples are needed to achieve a F_1 score above 0.800, while $F_1 > 0.750$ is achieved with just 3K annotated lyrics. Interestingly, the slope of the curve shows a rising trend when approaching the full size of the training

[28] https://www.rapitalia.it/testo/qui-e-selvaggio/primo-e-squarta/la-pietra-dello-scandalo-6421.
[29] https://testicanzoni.rockol.it/testi/vacca-la-colpa-68940016.

Table 3 Classification scores of LR-BOW , 1D-CNN$_{FT}$, and UmBERTo-CC$_C$ on the revised annotations on the 35 missing explicit labels

System	Non-explicit			Explicit			Macro			Weighted			A
	P^0	R^0	F_1^0	P^1	R^1	F_1^1	P	R	F_1	wP	wR	wF_1	
LR-BOW	.979	.959	.969	.547	.712	.619	.763	.835	.794	.951	.942	.946	.942
1D-CNN$_{FT}$	.968	.993	.980	.833	.531	.648	.900	.762	.814	.959	.962	.958	.962
UmBERTo-CC$_C$	.975	.990	.982	.813	.642	.717	.894	.816	.850	.964	.967	.965	.967

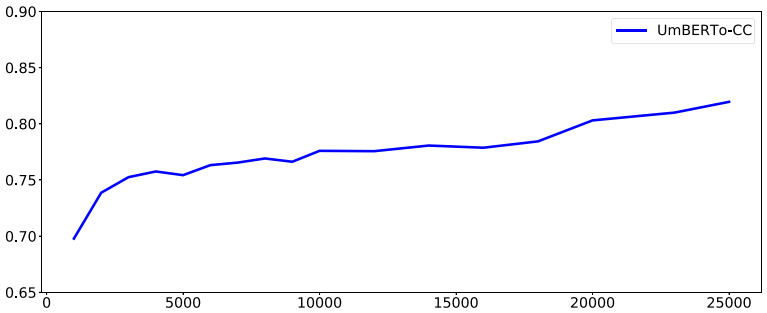

Fig. 1 The F_1 trend for UmBERTo-CC$_C$ obtained by varying the number of training samples

set, thus suggesting that (potentially) higher scores can be achieved providing more annotated lyrics.

5.2.2 Balancing the training dataset

When working with classification tasks in an unbalanced settings, where one class (in our case the non-explicit one) is much more populated than the other one (the explicit one), under/over-sampling balancing techniques (i.e., removing/replicating random samples in the training material) may contribute some performance improvements. Indeed, some previous works on explicit lyrics detection with English content (Bergelid 2018; Rospocher 2021) have reported some (marginal) performance improvement by applying some over-sampling of the less represented class (i.e., explicit lyrics). We decided to investigate if this holds also for detecting explicit Italian lyrics, experimenting with the best performing system on the orginal dataset (UmBERTo-CC$_C$). Therefore, we built 3 variants of the dataset via over-sampling (Batista et al., 2004), having different proportions of explicit / non-explicit lyrics (15%/85%, 33%/67%, and 50%/50% – a perfectly balanced scenario), and compared the performance of fine-tuning UmBERTo-CC$_C$ against the scores with the unaltered dataset (6%/94%). Each resulting model was evaluated against the same, original test set.

The resulting F_1 scores for each class, and their macro-averages, are reported in Table 4.

Table 4 Assessment of the impact of different level of balancing (over-sampling) on classification performance of UmBERTo-CC$_C$

System	Unaltered			15%/85%			33%/67%			50%/50%		
	F_1^0	F_1^1	F_1	F_1^0	F_1^1	F_1	F_1^0	F_1^1	F_1	F_1^0	F_1^1	F_1
UmBERTo-CC$_C$	.980	.658	.819	.978	.676	.827	.980	.697	.839	.977	.673	.826

The results show that by over-sampling the less represented class (explicit lyrics) it is possible to achieve a noticeable performance boost, resulting form a better prediction of the explicit class (c.f., substantial improvement of F_1^1) with almost no detrimental effects on the non-explicit lyrics detection (c.f., F_1^0 basically stable). Interestingly, among the tested dataset variants, the best scoring one correspond to the configuration where non-explicit lyrics double the explicit ones, and not when the two classes are perfectly balanced. This suggests that, potentially, a further classification improvement on this dataset may be achieved by finding the ideal proportion of explicit / non-explicit lyrics, something that, in a general setting, can be optimized on a validation set in a similar fashion as ML hyper-parameter tuning (e.g., using some dichotomy / grid search strategy for finding the optimal ratio).

6 Conclusions

We studied the problem of detecting explicit song lyrics written in Italian, i.e., determining if the lyrics of a given Italian song could be offensive or unsuitable for children. Inspired by previous works on English and Korean songs, where the problem was framed as a binary classification task, we assessed the performance of several machine learning classifiers, from competitive systems proved effective for the task (but on different languages) to more recent, state-of-the-art ones, based on neural language models.

The performance of all these classifiers was evaluated on a newly contributed dataset of Italian lyrics, consisting of ~34K songs annotated with explicitness information. The dataset was built using available resources (LyricsWiki, Spotify) and leveraging existing links between them.

The evaluation showed that the neural language model classifiers perform substantially better than other systems (best NLM classifier improves F_1 over non-NLM classifier by 0.040), with UmBERTo-CC$_C$, AlBERTo$_C$, and Italian-BERT$_C$ achieving the best scores. This somehow differs from the findings of previous work on another language (English - c.f., Fell et al. (2019)), where a state-of-the-art NLM model (BERT) performed less effectively than simpler models such as logistic regression. We believe this is an interesting finding of the work on the usage of NLMs for text classification, deserving additional assessment with other languages.

Moreover, we provided further analyses to complement the conducted evaluation. Besides discussing the computational aspects of the considered approaches and the quality of the prediction, we assessed the performance of the best scoring system

(UmBERTo-CC$_C$) in a few-shot learning scenario and when trained on balanced dataset. For the first analysis, we observed that good predictions ($F_1 > 0.75$) can already be achieved with as few as 3K training samples, although at least 20K annotated lyrics are needed to achieve performance quite close to the full dataset ones ($F_1 > 0.8$). For the second analysis, the results show that a tangible performance improvement can be obtained via over-sampling to reduce the balance difference between the classes, and finding the ideal proportion may require validating the system with varying ratios of explicit / non-explicit lyrics.

As future work, we plan to replicate similar analyses also on other languages, to understand whether the findings of our work are confirmed or counterfeited, or if it is possible to derive at least some general trends among similar languages. We will also consider a post-editing revision of the dataset driven by the classifiers, as discussed in Sect. 5, to further improve its quality.

Funding Open access funding provided by Università degli Studi di Verona within the CRUI-CARE Agreement.

References

Batista, G. E. A. P. A., Prati, R. C., & Monard, M. C. (2004). A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Exploration Newsletter, 6*(1), 20–29. https://doi.org/10.1145/1007730.1007735

Bergelid, L. (2018). Classification of explicit music content using lyrics and music metadata. Master's thesis, KTH, School of Electrical Engineering and Computer Science (EECS), http://kth.diva-portal.org/smash/record.jsf?pid=diva2%3A1228997&dswid=-4612

Bloehdorn, S., & Hotho, A. (2006). Boosting for text classification with semantic features. In B. Mobasher, O. Nasraoui, B. Liu, & B. Masand (Eds.), *Advances in web mining and web usage analysis* (pp. 149–166). Heidelberg: Springer.

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics, 5*, 135–146.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., & Amodei, D. (2020). Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, & H. Lin (Eds.), Advances in Neural Information Processing Systems, Curran Associates, Inc. (Vol. 33, pp. 1877–1901). https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf

Chin, H., Kim, J., Kim, Y., Shin, J., & Yi, MY. (2018). Explicit content detection in music lyrics using machine learning. In: 2018 IEEE International Conference on Big Data and Smart Computing,

BigComp 2018, Shanghai, China, January 15-17, 2018, IEEE Computer Society, pp 517–521, https://doi.org/10.1109/BigComp.2018.00085,

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning, 20*(3), 273–297. https://doi.org/10.1023/A:1022627411411

Devlin, J., Chang, MW., Lee, K., & Toutanova, K. (2019) BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1* (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota, (pp. 4171–4186). https://doi.org/10.18653/v1/N19-1423, https://www.aclweb.org/anthology/N19-1423

Fell, M., Cabrio, E., Corazza, M., & Gandon, F. (2019). Comparing automated methods to detect explicit content in song lyrics. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*. INCOMA Ltd., Varna, Bulgaria (pp. 338–344). https://doi.org/10.26615/978-954-452-056-4_039. https://www.aclweb.org/anthology/R19-1039

Fix, E., Hodges, J., & of Aviation Medicine US (1951) Discriminatory Analysis: Nonparametric Discrimination, Consistency Properties. USAF School of Aviation Medicine, https://books.google.it/books?id=VN07ngEACAAJ

Joulin, A., Grave, E., Bojanowski, P., & Mikolov, T. (2017). Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. Association for Computational Linguistics, Valencia, Spain (pp. 427–431). https://www.aclweb.org/anthology/E17-2068

Kibriya, A. M., Frank, E., Pfahringer, B., & Holmes, G. (2004). Multinomial naive bayes for text categorization revisited. In *Proceedings of the 17th Australian Joint Conference on Advances in Artificial Intelligence*. Springer, Berlin, AI'04, p 488–499, https://doi.org/10.1007/978-3-540-30549-1_43,

Kim, J., & Yi, MY. (2019). A hybrid modeling approach for an automated lyrics-rating system for adolescents. In L. Azzopardi, B. Stein, N. Fuhr, P. Mayr, C. Hauff, & D. Hiemstra (Eds.), Advances in Information Retrieval—41st European Conference on IR Research, ECIR 2019, Cologne, Germany, April 14–18, 2019, Proceedings, Part I, Springer, Lecture Notes in Computer Science, vol 11437 (pp. 779–786). https://doi.org/10.1007/978-3-030-15712-8_53,

Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar (pp. 1746–1751). https://doi.org/10.3115/v1/D14-1181, https://www.aclweb.org/anthology/D14-1181

Le, Q., & Mikolov, T. (2014). Distributed representations of sentences and documents. In: Proceedings of the 31st International Conference on International Conference on Machine Learning—Volume 32, JMLR.org, ICML'14, p II–1188–II–1196

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE, 86*(11), 2278–2324. https://doi.org/10.1109/5.726791

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). Roberta: A robustly optimized BERT pretraining approach. CoRR abs/1907.11692, http://arxiv.org/abs/1907.11692

McNemar, Q. (1947). Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika, 12*(2), 153–157.

Polignano, M., Basile, P., de Gemmis, M., Semeraro, G., & Basile, V. (2019). Alberto: Italian BERT language understanding model for NLP challenging tasks based on tweets. In R. Bernardi, R. Navigli, & G. Semeraro (Eds.), *Proceedings of the Sixth Italian Conference on Computational Linguistics, Bari, Italy*, November 13-15, 2019, CEUR-WS.org, CEUR Workshop Proceedings, vol 2481, http://ceur-ws.org/Vol-2481/paper57.pdf

Rokach, L. (2010). Ensemble-based classifiers. *Artificial Intelligence Review, 33*(1–2), 1–39. https://doi.org/10.1007/s10462-009-9124-7.

Rospocher, M. (2021). Explicit song lyrics detection with subword-enriched word embeddings. *Expert System Application, 163,* 113749. https://doi.org/10.1016/j.eswa.2020.113749.

Schweter, S. (2020). *Italian bert and electra models.*https://doi.org/10.5281/zenodo.4263142

Sparck Jones, K. (1988). *A Statistical Interpretation of Term Specificity and Its Application in Retrieval* (pp. 132–142). GBR: Taylor Graham Publishing.

Vanni, L., Ducoffe, M., Aguilar, C., Precioso, F., & Mayaffre, D. (2018). Textual deconvolution saliency (TDS) : a deep tool box for linguistic analysis. In *Proceedings of the 56th Annual Meeting of the*

Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, Melbourne, Australia, pp 548–557, https://doi.org/10.18653/v1/P18-1051, https://www.aclweb.org/anthology/P18-1051

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Lu., & Polosukhin, I. (2017). Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems*. Curran Associates, Inc., Vol 30. https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., & Hovy, E. (2016). Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, San Diego, California (pp. 1480–1489). https://doi.org/10.18653/v1/N16-1174, https://www.aclweb.org/anthology/N16-1174