Using Parallelized Neural Networks to Detect Falsified Audio Information in Socially Oriented Systems

Sergiy Yakovlev [1], Artem Khovrat [2] and Volodymyr Kobziev [2]

[1] *Kharkiv National University of Radio Electronics, 14, Nauky, Ave., Kharkiv, 61166, Ukraine*
[2] *Lodz University of Technology, 90-924 Lodz, Poland*

Abstract
The problem of information counterfeiting with technology development is becoming more acute and can provoke social unrest. Modern algorithms can automatically create videos, audio, and text data, and more people are not able to reveal the fact of deception in them. This is especially true of audio information in socially oriented systems. Different mathematical models and concepts can be used to counteract this problem. The most effective of these are derivatives of convolutional and recurrent neural networks. However, the complexity of the data and its volume determine the problem of slowness of the classification. One of the approaches to solving this issue is the use of parallelism. With the framework of the current study, given the simplicity of monitoring, it was decided to focus on MapReduce technology. In the course of this investigation, the features were characteristic of audio information both in its text form and in the form of a signal. The possibilities for the rapid process of data and their augmentation using vector autoregression algorithms are determined. Based on the analysis of international experience, it was decided to use hybrid neural networks based on convolutional and recurrent networks. Several experiments were conducted to determine the effectiveness of the proposed approaches and the expediency of using MapReduce. The results obtained suggest that the most effective algorithm is the combination of two selected architectures and that it is advisable to use the specified parallelization technology to accelerate the selected models. This fact shows the possibility of practical application of the proposed approach to detecting falsified audio.

Keywords [1]
Classification, fake information, natural language processing, neural networks, parallelization, socially oriented systems

1. Introduction

In recent decades, technologies capable of falsifying information have reached the level where the need to detect forgeries in socially oriented systems is being discussed at the legislative level [1]. At the same time, the degree of acuteness of the problem is diversified depending on the type of relevant data. In particular, in the case of video information, the distortion has not yet reached the required level [2]. In the case of text and photos, there are already quite thorough studies and even certain implementations of them to determine the fact of forgery [3, 4]. At the same time, audio falsification has only recently been able to go beyond simple identification, that is, using human hearing. This state of affairs allows ordinary citizens to confuse a fake record with a real one.

In normal conditions, the problem can give rise to local conflicts in a group of people, this is especially noticeable in social networks [5]. The situation becomes more acute in conditions of military and political instability when any information is perceived through the prism of intensified emotions, which slow down the process of critical thinking. In the case when fakes are part of the news information field, they can accelerate social shifts caused by the crisis and thus intensify its

Information Technology and Implementation (IT&I-2023), November 20-21, 2023, Kyiv, Ukraine
EMAIL: sergiy.yakovlev@p.lodz.pl (A. 1) artem.khovrat@nure.ua (A. 2); volodymyr.kobziev@nure.ua (A. 3);
ORCID: 0000-0003-1707-843X (A. 1); 0000-0002-1753-8929 (A. 2); 0000-0002-8303-1595 (A. 3);
© 2023 Copyright for this paper by its authors.
Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).
CEUR Workshop Proceedings (CEUR-WS.org)

CEUR-WS.org/Vol-3624/Paper_19.pdf

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

consequences [6]. It can be economic, purely social, or even military in nature, negatively affecting the mood of the population. As an example of a similar situation: fake audio related to the COVID-19 pandemic [7], or a huge number of falsified recordings at the beginning of the invasion of the Russian Federation into the territory of Ukraine [8], which were used to hide the facts of violations of the laws of war or discredit the Armed Forces of Ukraine. It should be noted that, although the possibilities for high-quality forgery of audio data appeared relatively recently. They are already much easier to implement, unlike a photo, which requires a large amount of source material and a sufficient amount of time. This, in turn, only increases possible risks for society and the state.

The state of affairs became the basis for the fact that more and more countries are discussing the fight against such information, although, mostly, they pay attention only to its textual component. Instead, the issue of audio alteration detection is quite open, although partially addressed, especially when it comes to the addition of real information, rather than synthetic generation. Therefore, it was decided to apply known methods of determining the fact of forgery of text or photo data for any type of audio. Based on world practice, neural networks, in particular convolutional and recurrent, are the most effective. However, it should be noted that they require significant amounts of initial information as well as time for training the model. To solve these problems, you can use the principles of augmentation of audio materials and parallelism, respectively. However, classical forms of parallelization do not guarantee a significant gain in speed during model training [9]. As a basic alternative to these principles, MapReduce technology is used, which, in addition to training, allows you to speed up the process of determining the fact of falsification. As part of the current work, a review of models of fake audio classification based on neural networks and the possibility of modifying these algorithms and means of their parallelization will be carried out. The purpose of the work: development of an effective model for determining the fact of falsification of sound data, using MapReduce technology. To achieve this goal, the following list of tasks has been defined:

– determine the features of audio in socially oriented systems;
– analyse the international experience of determining the fact of falsification for various types of information;
– based on the conducted analysis, form a set of limitations and assumptions, and define models that will be used in further research;
– carry out a description of algorithms that would allow the reprocessing of audio information both in the form converted to text and in the form of a signal;
– review selected architectures of neural networks and determine their main hyperparameters;
– reveal the essence of MapReduce technology and describe the implementation of the proposed approach;
– form an experiment plan and a multi-criteria selection task that would allow determining the expediency of using MapReduce and choosing the most effective classification algorithm;
– analyse the results of the experiment and form appropriate conclusions.

2. Domain analysis

First of all, it should be noted that the audio can be falsified in different ways, in particular:

– synthetic creation of audio with the help of artificial intelligence: takes place in the presence of a large amount of source material and the need to obtain the voice of a specific person;
– composition of existing sound tracks to distort the essence of the original information: unlike the previous case, generation algorithms may not be used here, but the need to obtain the voice of a specific person remains;
– contextual distortion: if the owner of the voice on the audio is not important, fake news or announcements may be recorded on the audio tracks.

Regardless of the nature of the falsification, all cases focus on changing the context to obtain the desired result: deterioration of the mood of the population, blackmail, etc. With this in mind, the following list can be made of how falsified audio can be detected and classified based on the information in the middle of the message:

– using an unnatural number of rhetorical questions (contextual distortion of socially important information). Conducted linguistic studies indicate that in the official, business, and journalistic styles to be used by mass media, this type of speech construction is rarely used [10]. This feature is typical for both text news and audio and video;

– absence of negative constructions to reduce cognitive load in combination with the pessimistic coloring of the chosen words. As an example, the replacement of the word "problem" with "catastrophe" can be given. However, it should be noted that during oral speech, the use of profanity does not allow one to determine the nature of the color with certainty, therefore, for further analysis, the evaluation of such words will be formed based on the context;

– using appeals and incentives in inappropriate contexts. It should be noted that in the case of contextual distortion, which aims to replace real news, such constructions immediately indicate the falsity and incorrectness of the specified information. However, when looking at mimic recordings, these features may be characteristics of the speech process of the attacker's target;

– use of an unreasonable number of pronouns. This factor mostly has room for contextual distortion, which imitates the journalistic style of presentation.

The specified characteristics are not exhaustive, but it should be emphasized that in the case of processing of audio recordings transformed into text form, some features may not manifest themselves. In particular, it is known that when creating fake news, short sentences and words are often used, or they contain a large number of various errors [11]. These and similar characteristics will not be taken into account further, since they may appear due to incorrect audio recognition or generally be a special part of the speech process of people present on the audio recording (for example, when mixing two languages or using regional dialects). These same features can lead to receiving a greater number of false positive cases when detecting fakes.

In general, the field of determining the falsity of information is not new, as already noted above. In the study of fake text news by several groups of Spanish scientists [12, 13], it was shown that machine learning by its very nature requires a sufficiently large amount of information to achieve a positive (accuracy of more than 95%) classification result, besides, it is quite sensitive to data outliers. However, taking into account publicly available databases, the specified shortcomings are not significant, as was shown in the work of Ukrainian researchers [14]. A similar justification can be applied to other types of information, including audio. This was demonstrated in their work by researchers from the Massachusetts Institute of Technology [15].

Regarding other methods of detecting falsified information, another quite popular method is the creation of graph models, which was extensively studied by scientists from Harvard [16] on the example of fake accounts and guarantees a quick result with minimal basic data. However, when applied to audio or text information, the method will require significant refactoring and thus the gain in speed is leveled off. For the issue of detecting fake data it is worth mentioning the problem of detecting spam. A group of Chinese-American scientists proved the possibility of effective application of Markov networks [17]. However, given the specificity of the area, their application is quite cumbersome and will require significant computing power, which was studied by Canadian scientists from Montreal [18]. When considering visual information, such as a photo or video, you can apply autoregressive models to detect deviations from the basic values, in other words, to detect the fact of data manipulation. This approach was successful.

3. Mathematical representation

When considering audio information, there are two main approaches of processing – considering the data as a signal and considering it as a text record. Start the presentation of the proposed approach from the second approach.

3.1. Analysis of audio as text

Based on the analysis of the work of a group of Chinese scientists, it was determined that the creation of an own Speech to Text module is accompanied by the following problems:

– quality of recordings for training;

– lack of data for model formation (the problem is especially acute for languages with small corpora);

– ignoring pronunciation defects;

– correct processing of dialectics, neologisms, abbreviations, etc.

The specified list is not complete, so in order to avoid the specified problems and achieve the best result of converting audio to text, it was decided to use Google Speech to Text, in particular, its appropriate wrapper for the Python3 programming language. Additionally, with the help of voices recorded by 20 people from different regions of Ukraine and with various speech disabilities, as well as 20 recordings from Ukrainian-language films, it was established:

– the system has limited capabilities in recognizing abbreviations;

– if the pauses between words are too long, the module will consider them as separate sentences;

– the quality of recordings does not significantly affect the quality of recognition: the presence of additional noise is leveled at the expense of the audio remastering stage;

– without taking into account the above, the accuracy of recognition reached more than 95%, the exceptions were the rural dialects of the western and eastern regions.

These statements are limitations of the current work.

To convert the received text information into a numerical representation, the following algorithm will be used:

– break the text into sentences and separate words;

– remove words without a significant lexicographic load and those that do not affect the results of the algorithms (so-called stop words). For example, "however", "this", "or", "etc";

– form a text dictionary based on the received words;

– separate the bases of each word in the dictionary and remove repetitions (perform a stemming operation);

– define a lemma for each word in the dictionary and remove repetitions (perform lemmatization);

– determine the frequency characteristic of each word (its description will be made below) and its emotional colouring using Sentiment Analysis tools built into the nltk model of the Python3 programming language;

– modify the assessment of emotional colouring within each separate sentence based on the rules established earlier;

– aggregate and normalize the frequency-emotional indicator for each sentence, it will serve as a target indicator for further use of neural networks;

– find the index of suspiciousness in a normalized form, based on a list of words that are often used in fake information.

In addition to the indicators, the following will also be used as input values:

– frequency-emotional characteristics of the 50 most popular news for the date of creation of the audio recording. This indicator will allow to take into account the external news environment and adjust the classifier's assessment accordingly;

– message weight. It was decided to create a data set in which the audio will be divided into 4 groups: fragments of home dialogue, general news, information from the place of emergency events, news of special importance. Marking will be carried out from 1 to 4, respectively. The value of the weight indicator will also be normalized;

– degree of reliability of audio to text conversion. It will be determined by comparing the actual text of the message with what was processed by Google Speech to Text, as the ratio of correctly recognized words.

Regarding the frequency response, it was decided to use BM25, which is a certain modification of TF-IDF. The purpose of TF-IDF is to count the importance of each word in the query and text, considering the frequency of use of the term both in a particular document and in the corpus as a whole. Conventionally, the word "and" may be one of the most used in a particular sentence, but it is often used in the corpus as a whole, so it will have less significance when searching this corpus. The method is based only on statistics and is calculated quite quickly, so it is still popular for problems where more complex solutions are not required. In the case of BM25, the saturated term frequency is added to the TF. That is, if the term already has a high frequency, then after a certain mark the increase in frequency will not have a significant effect on the TF estimate. The IDF is used in the same way. In addition, two parameters, parameters k1 and b, are used, which can be adjusted for a specific case. k1 is responsible for frequency saturation, and b is for the measure of the influence of the length of the document on the results.

The choice of TF-IDF was determined by the results of previous studies devoted to the analysis of textual news [21]. Inaccuracies and limitations reduced the effectiveness of the proposed classification methods. In particular, one of the problems was the oversaturation of certain terms that cannot be considered stop words, for example, "catastrophe".

3.2. Analysis of audio as signal

The first step in preparing audio for processing as a signal is to separate the vocalized part of the signal from the silence. Such an operation is necessary, because the very first part contains the key elements of human speech. One of the commonly used ways of marking an audio signal is to divide it into three states:

– area of silence (S), where there is no pronunciation;
– non-vocalized region (U), where the resulting waveform has an aperiodic or random character (occurs when the vocal cords do not vibrate);
– vocalized region (V), where the resulting waveform is quasi-periodic (occurs when the speaker's vocal cords are tense and, accordingly, vibrating).

Regarding the combination of the first two areas indicated above, it is worth noting that there are indeed methods that would allow us to separate silence from non-vocalization, but they require constant reconfiguration for different environments, which in the context of audio news is an inefficient procedure. A similar problem exists for methods that separate the vocalized region from others based on low energy. Therefore, taking into account the statement and the fact that the noisy environment of news can vary, but is relatively stable, it was decided to apply methods based on distribution. At the same time, the signal has a Gaussian nature. Thus, to isolate the necessary part of the audio, you can use the Mahalanobis distance function, which is a linear pattern classifier (LPC).

To determine the parameters of the Gaussian distribution, it is necessary to determine the base window. For this purpose, it was decided to apply the method of expert evaluation. 30 sound processing specialists from Kharkiv, Kyiv, Dnipro and Vienna were interviewed. It is established that the optimal window size is 200 ms. The Gaussian distribution for the one-dimensional case is determined via the following formula:

$$g(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{(x-\mu)^2}{2\sigma^2}}, \tag{1}$$

where μ – mean, σ – standard deviation.

Given the above, the following set of rules can be defined:

$$\begin{aligned} P\left[|x-\mu| \le \sigma\right] &\approx 0.68, \\ P\left[|x-\mu| \le 2\sigma\right] &\approx 0.95, \\ P\left[|x-\mu| \le 3\sigma\right] &\approx 0.997. \end{aligned} \tag{2}$$

The Mahalanobis distance can be determined as follows using the formula:

$$r = \frac{|x-\mu|}{\sigma}. \tag{3}$$

Taking into account (2) and (3), it can be established that with a probability of 99.7% the distance will be less than 3.

The process will take place as follows:

– the algorithm performs a gradual analysis of audio in a window of 200 ms and determines the standard deviation and the average value;

– for each subsequent window, the Mahalanobis distance is calculated using the previously obtained values;

– if the distance exceeds 3, then the sample to be vocalized, otherwise, it can be replaced with empty values, actually deleting it.

The neural network will receive the transformed audio as input, but this time the sampling window will be determined by cross-validation. As mentioned above, to avoid the problem of lack of audio signals, it was decided to carry out the augmentation process using autoregression. Formally, it can be submitted as follows:

$$\Phi_0 y_t = \Phi_1 y_{t-1} + \cdots + \Phi_p y_{t-p} + \Theta_0 u_t + \Theta_1 u_{t-1} + \cdots + \Theta_q u_{t-q}, \quad (4)$$

where y_t – N-dimensional time series; Φ_i, Θ_j – non-degenerate matrices of autoregression coefficients of dimensionality N×N, $i = \overline{1, p}$, $j = \overline{1, q}$.

Here it is worth noting that since the matrix of coefficients is not degenerate, it is easy to normalize them in the range from 0 to 1. Model (4) can be used in the process of considering short-term periods, in addition, it is necessary to guarantee the absence or insignificance of external influence. Since within the scope of this work, it was decided to consider the medium-term perspective, it is necessary to find the delta between (4) and the forecast for the previous period$\Phi_0 y_{t-1}$:

$$\Phi_0 \Delta y_t = \Pi y_{t-1} + \Psi_1 \Delta y_{t-1} + \ldots + \Psi_{p-1} y_{t-p+1} + \Theta_0 u_t + \Theta_1 u_{t-1} + \ldots + \Theta_q u_{t-q}, \quad (5)$$

where $\Pi = -(\Phi_0 - \Phi_1 - \ldots - \Phi_p)$; $\Psi_i = -(\Phi_{i+1} + \ldots + \Phi_p)$; $i = \overline{1, p-1}$.

The resulting matrix of coefficients, subject to the total number of unknowns that must be taken into account during generation, may vary depending on the selected model. Within this study, the following variations will be considered:

– simple vector autoregression;

– seasonal vector autoregression;

– vector autoregression of the distributed lag;

– vector autoregression of the moving average;

– vector autoregression of the integrated moving average.

In this case, the accuracy of data augmentation can be considered the only factor of effectiveness. In fact, it is necessary to ensure that the distribution between vocalized and unvocalized 200ms samples does not change.

3.3. Architectures of neural networks

The chosen BiLSTM and CNN-RNN architectures are inherently descendants of classical convolutional and recurrent neural networks.

RNN consists of several hidden layers that work one after the other. At the same time, each subsequent layer at the input receives the result of the work of the previous one. The specified feature is called short-term memory by analogy with the human brain.

Schematically, the architecture of a recurrent neural network can be presented in the form shown in Figure 1.

Inside the hidden layer, the output data is gradually processed using gradients. However, if the data is specific and is not inherently bounded (either above or below), exploding or vanishing gradient problems may arise. These are situations when the gradient value starts heading towards infinity and 0, respectively. Based on international experience, when analyzing text information (and signals as well), this problem can occur. To overcome this shortcoming, it was decided to use neural networks with support for short-term and long-term memory (LSTM).

The essence of the mathematical apparatus in LSTM consists in the gradual use of several sigmoids and hyperbolic tangents, which allow you to adjust the values in such a way as to avoid the direction both to 0 and to infinity.

At the first stage, Forget Gate adds two input weight data and multiplies by the activation sigma function. The range of values of this function limits the result of the execution of the specified step in the range from 0 to 1. After completion, the result is multiplied by the data from the long-term memory channel.

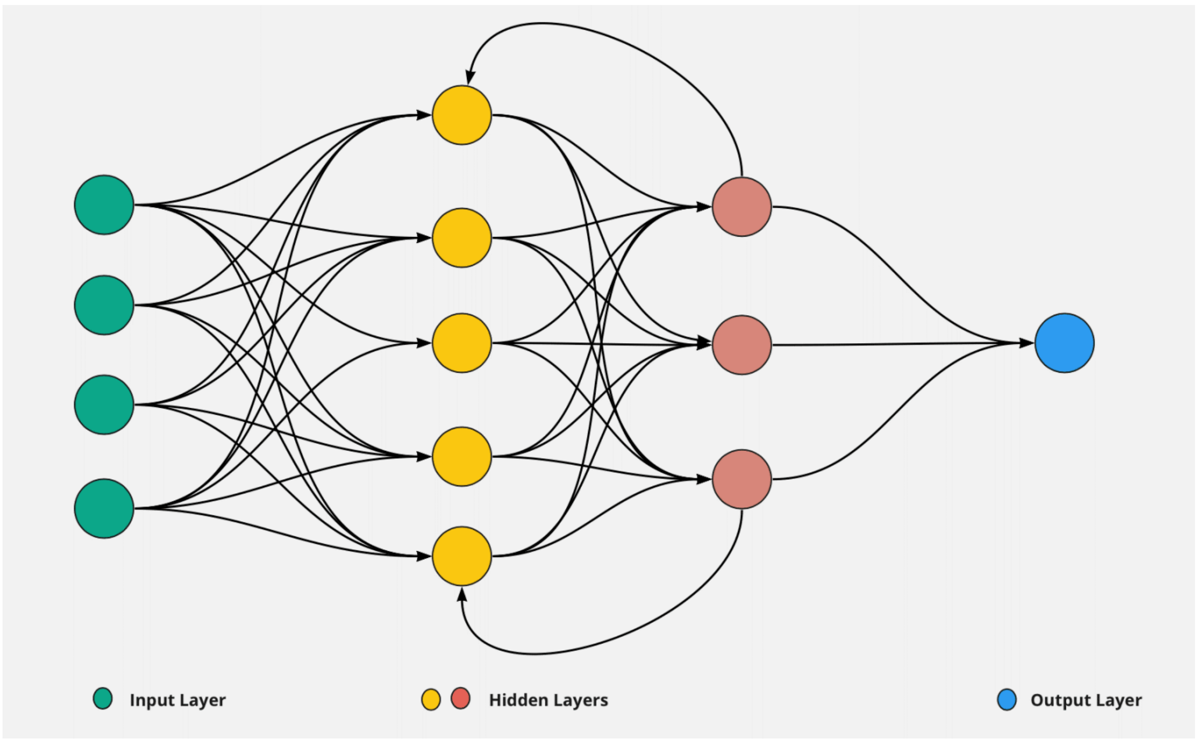


Figure 1: Scheme of the RNN architecture [22]

The second stage is the Input Gate, which performs a similar multiplication by a sigma function. However, this time the result is corrected by applying a hyperbolic tangent. Such an operation allows you to eliminate the problem of going to infinity when adding to a value from a long-term memory channel.

As a result of the operation of the two specified stages, a state of memory is formed, which, together with the input and previous output data, serves as the basis for the formation of a new value of short-term memory. This stage is called Output Gate and is the final step of one hidden layer. The essence is to find the hyperbolic tangent from the long-term memory value, which is then multiplied by the sigma function from the previous value in the short-term memory.

The steps described above can be presented as shown in Figure 2.

Although this architecture avoids the problem of gradients, it still has one significant drawback when processing natural language – the impossibility of taking into account the future context.

Let us have the beginning of the sentence "Apple is something that...". A conventional LSTM architecture will not be able to determine what exactly is meant by "Apple" - a fruit or a company, because it does not have information about the end of this sentence. For her, the options are "Apple is something that competitors simply cannot reproduce." and "Apple is something that I like to eat." are idempotent. To avoid this problem, it was decided to use a bidirectional recurrent neural network with support for long- and short-term memory (BiLSTM). The essence of this network is the combination of two LSTMs that are directed in different directions. In this case, the "auxiliary network" allows you to take into account the context for the beginning of the sentences.

After working out two subnets, the result of both levels is combined, first by simple concatenation, and then by linear transformations. To determine the relevant operations. It was decided to conduct cross-validation. During which it was established that the best result is achieved when using averaged values. A similar conclusion was obtained during an expert evaluation among 10 people engaged in natural language processing. Schematically, the architecture of a bidirectional recurrent neural network with long-term and short-term memory support can be presented in the form shown in Figure 3.

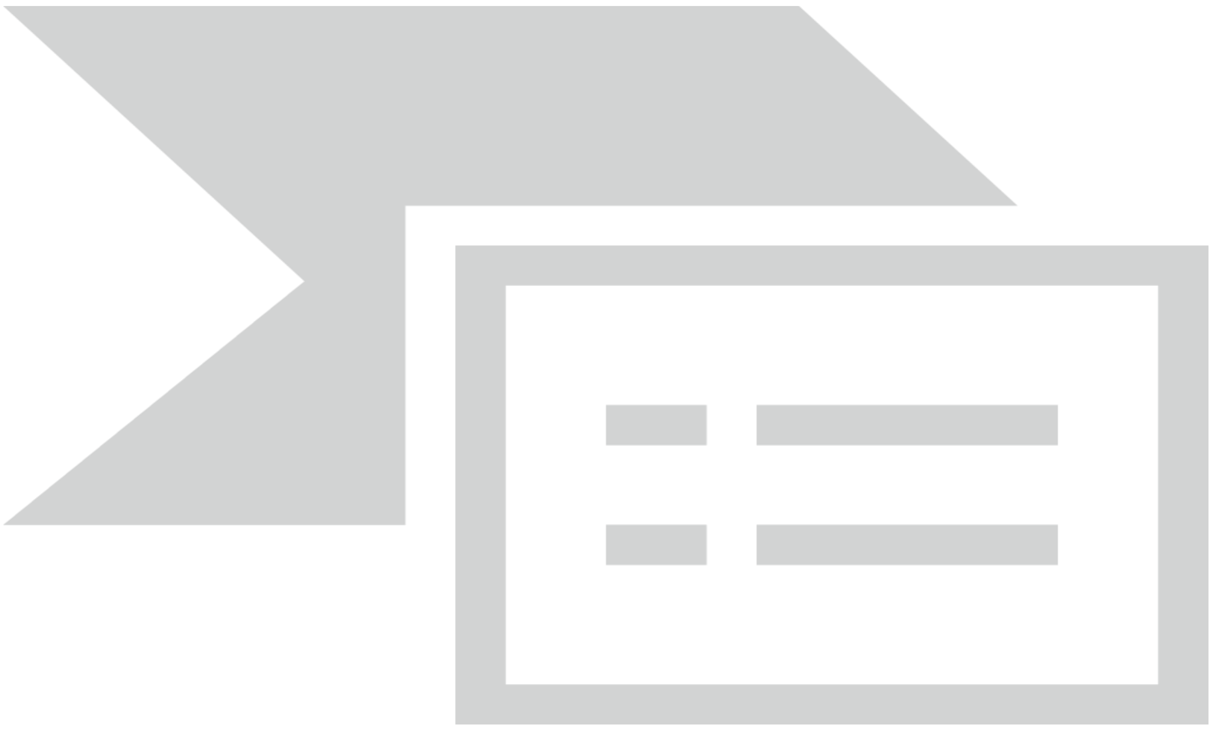


Figure 2: Hidden Layer of the LSTM neural network architecture [23]

Figure 3: Scheme of the BiLSTM architecture [23]

Compared to the previous case, CNN-RNN architecture has neither short-term nor long-term memory. Instead, it uses a convolution layer, which allows you to significantly reduce the dimensionality of the output data. It is especially effective in pattern recognition and determining the fact of falsification of images or videos.

In order to be able to build the most effective CNN architecture, it is necessary to set a number of hyperparameters of the model. One of the most important among them is the size of the filter. This is such an element of the hidden layer that performs data traversal and convolution. After cross-validation, it was determined that the 5×5×5 dimension filter is best for the chosen case.

Here it is worth noting that the last value of the dimension in our case will correspond to the number of descriptors that make up the target variable. That is why for the analysis of audio as text, the dimension will be 5×5×5, but in the case of analysing audio as a signal, authors will consider only the standard deviation and the average value as descriptors, so the dimension of the filter will be 5×5×2.

When it passes through the data, there is a scalar product between the filter entries and the input information. This will make it possible to form an activation map, the size of which will be equal to the number of used filters - in other words, the depth of the neural network. Based on the number of factors that will be taken into account during classification, it was decided to stop at a depth of 5 and 2, respectively

In addition to the specified hyperparameter, the following characteristics are considered for the current architecture:

– size of the kernel (during cross-validation, a kernel ranging from 2 to 5 was used and the optimal value was set to 4);

– step size during consideration. Based on recommendations that indicate the undesirability of using a step greater than 3 for text, an optimal value of 1 was determined;

– based on the set step, the option of adding insignificant zeros will not be applied;

– based on the specifics of the subject area, it was decided not to apply the displacement parameter.

It should be noted that the number of convolutional network layers for textual information analysis must be equal to 1

The schematic architecture can be represented as shown in Figure 4.

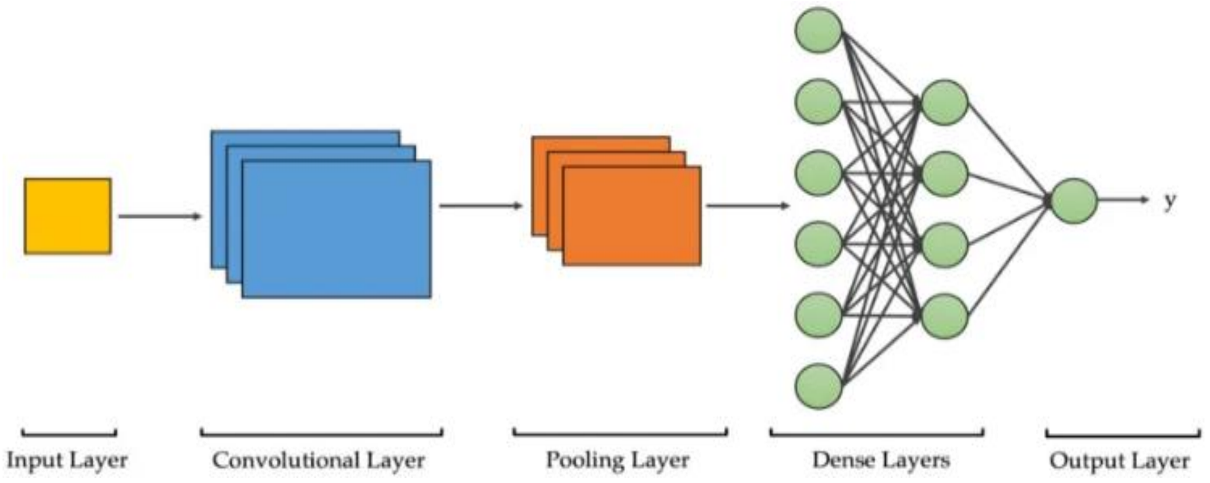


Figure 4: Scheme of the CNN architecture [24]

The problem of the specified architecture when using it for processing natural languages is the limitation in taking into account the context. Usually, passing the filter allows you to take into account the context of each word, but the specificity of the Ukrainian language lies in large sentences. Thus, the salient context may be outside the CNN model's filter. To avoid this problem, it was decided to combine RNN and CNN. Although there are several ways of such a combination, the current work will consider only the RCNN architecture, which sequentially uses two neural networks. In other words, after convolution, the result is not only concatenated, but sent to the recurrent neural network layer.

To avoid the above-mentioned problems when considering the text, it was decided to use not the classic RNN architecture, but the above-mentioned bidirectional recurrent neural network with support for long- and short-term memory. Thus, the RCNN architecture can be presented in the form shown in Figure 5. As mentioned above, the use of complex neural networks has a significant drawback – the time of their training and processing. In addition, data processing time is also a problem. In order to reduce the impact of these shortcomings, it was decided to use the MapReduce technology.

3.4. MapReduce technology

The MapReduce technology consists in dividing the original data set in such a way that it is processed on separate nodes. The key operations are the use of mapping and reduction functions. The first allows you to distribute data between the nodes on which the desired processing is carried out, and the second instead collects data from all nodes and unifies them. It is worth noting that the MapReduce technology defines only the specifics of the implementation of the corresponding modules within certain frameworks. At the same time, in general, this implementation can differ significantly. For this paper, it was decided to use MapReduce, which is offered by Hadoop [26]. Graphically, the proposed solution can be presented as shown in Figure 6.

Figure 5: Scheme of the RCNN architecture [25]

For the above case, special attention should be paid to the distribution and combination functions. They are necessary to implement additional parallelization in each of the nodes, using different memory regions. To better understand the essence of this approach, you can consider basic nodes as processes, and specified memory regions as streams.

In addition to these two functions, an important feature is the sorting of information before reduction. The current work considers data that significantly depends on the order and at the same time does not have additional time labels, such as in time series. To avoid the problem during reduction, it was decided to add field with the sequence number of each fragment of text/signal. It will be used for sorting.

Regarding the peculiarities of the implementation of the proposed approach, it is worth noting. MapReduce will be used independently in the preprocessing of the raw data and during the training of neural networks. In the case of signals, it is only important to carry out the reduction in the correct order to carry out the reduction. To process audio as text, it is necessary to take into account the importance of forming as large a vocabulary as possible. For this purpose, it was decided to create a separate non-relational database with multithreading support, where the entire existing dictionary will be recorded after basic processing. Thus, the more material will be processed, the higher the accuracy of the formation of the corresponding frequency characteristic will be.

For the RCNN architecture, the first step is the CNN layer. It iteratively adjusts the weights by computing their partial gradients after each set of training data is propagated through the network. Thus, parallelization during the training phase can be achieved by dividing the data into several fragments. Each piece of data is then fed into multiple CNNs, and each CNN is trained independently in parallel. The results are then aggregated using a reducer to produce the final results, which are then used to update the weights for the next iteration. After the CNN layer is finished, the aggregated data is transferred to BiLSTM. To speed up the bidirectional neural network, you can distribute the work of two neural networks between two nodes. In this case, the reduction function will serve as an aggregation function of the results of the two networks. Among the general advantages of the proposed approach, one can highlight its scalability, relative cheapness, ease of use, and the possibility of execution monitoring using the appropriate Hadoop tools (if internal monitoring is necessary, basic methods of the Python programming language are used).

Among the disadvantages, authors can single out the need to create a large amount of software code, the stealth of processing (although with the possibility of viewing log files) and the need for long-term configuration settings.

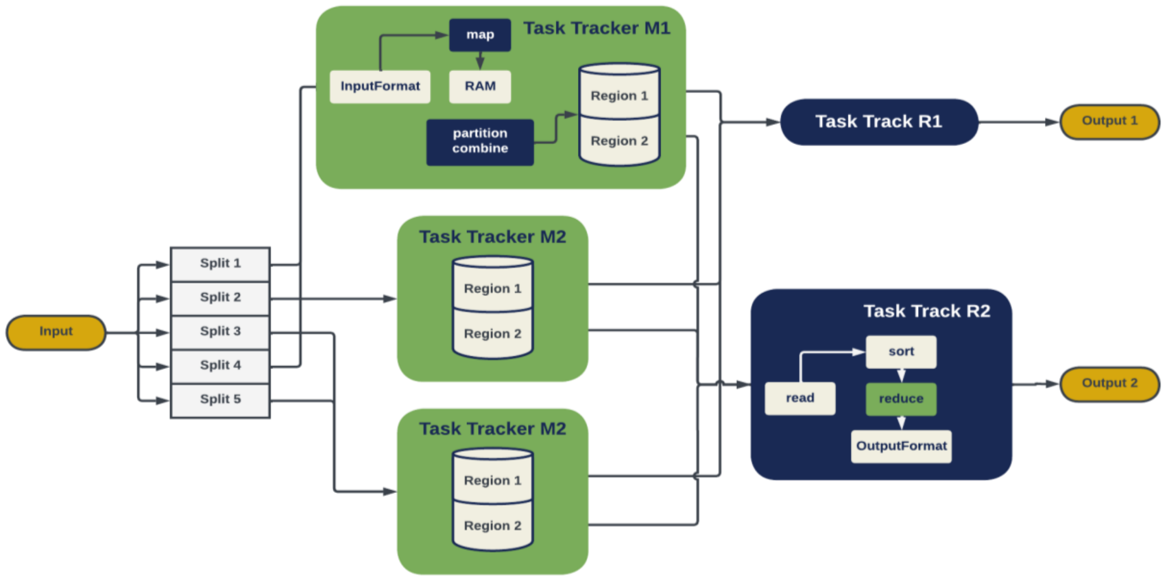


Figure 6: MapReduce in Hadoop

4. Experimental environment

In this article, the experimental environment refers to the task of multi-criteria selection to determine the most efficient classification method and the experimental design describing the data samples and MapReduce setup features.

4.1. Plan of the experiment

Given the specificity of the proposed research, the method of a controlled experiment was chosen. The base runtime has the following set of characteristics:

– CPU: Intel Core i5-1135G7;
– RAM: 16 Gb;
– VRAM: 4 Gb;
– OS: Ubuntu 21.04.

The specified characteristics were almost completely duplicated on virtual nodes, on which the process of partial calculation is planned to be carried out (RAM was reduced from 16 to 8). Their number will be from 3 to 4 (2 in the case of parallelization of bidirectional neural networks).

The datetime library for Python 3, which is accurate to the nanosecond, was chosen as the means of calculating the execution time. So that basic calculations do not slow down the program, it was decided to use the numpy and polars libraries. For the processing of natural languages (taking into account lemmatization, tokenization, and other necessary functions), it was decided to choose the python version of the nltk library. To implement neural networks, it was decided to use tensorflow using the tools provided by the pipeline submodule.

When considering audio in the form of text, as already mentioned, it was decided to use Google Cloud Platform to avoid the influence of the environment on the speed of work. The optimization of this tool made it possible to reduce the delay time when receiving text information decoding from 10 seconds (for processing 2 minutes of audio) to 2 seconds. The first value is obtained with the help of a self-built speech recognition tool.

As data for verification, authors will use a set of audio generated by our own hands, generated based on text news and partially modified in various ways, described earlier.

The first set of data concerns the full-scale invasion of Russia on the territory of Ukraine and consists of both simple news and reactions to certain events by users of social networks, television experts, etc. It was decided to dedicate the second data set to the election process in 2019, which was accompanied by the appearance of a large amount of falsified information. Each of these samples will be split 80 to 20 into training and test subsamples, respectively.

4.2. The problem of multi-criteria selection

To be able to compare the mentioned different types of neural networks with and without the use of MapReduce, it is necessary to decide on the main selection criteria. Given that the task of classification for socially acute processes is considered, the most important criteria are time-saving and accuracy of classification.

In general, the following list of factors was chosen:

- accuracy indicator;
- saving time of model training at the same capacity;
- saving time of data processing;
- saving the minimum permissible amount of data to achieve Accuracy = 90%;
- the possibility of taking into account the context.

Here it is worth noting that the data reprocessing time-saving indicator is important only for determining the autoregressive algorithm and the speed gain provided by the MapReduce technology. Therefore, only four metrics from the above will be used to evaluate the performance of neural networks.

Model training time savings will be measured in seconds using the above library. At the same time, authors do not limit the indicator itself.

To reduce the impact of possible measurements caused by problems with the accuracy of the time modules or the environment, it was decided to conduct 5 measurements for the time indicators and check the accuracy of the prediction on two data samples.

Classification accuracy is determined using a combination of F1-score and Precision, normalized in the range from 0 to 1. Authors will measure accuracy for two samples and take the average value.

Given that context can be taken into account to varying extents, as demonstrated by the example of LSTM and BiLSTM architectures:

- the context is fully taken into account in both directions - 5 points;
- the context is taken into account only in one direction - 4 points;
- the context is taken into account only in the neighborhood in both directions - 3 points;
- the context is taken into account only in the neighborhood in one direction - 2 points;
- the context is not taken into account - 1 point.

It should be noted that this indicator will not be used when classifying signals.

To determine which of the models is the most effective according to the above criteria, it was decided to apply the principle of linear additive convolution with weighting coefficients. To determine the weighting coefficients, it was decided to conduct an expert evaluation among journalists and analysts (the number of reviewers was 50 people).

In the issue of classification, both signals and text, the most important is the accuracy indicator. In second place are opportunities to take into account the context, and in third place are time indicators. In this way, the next rules can be assigned:

- for accuracy: 16 points;
- for the possibility of taking into account the context: 10 points;
- for saving time of model training: 2 points;
- for saving the minimum amount of data to achieve the required accuracy: 2 points.

With this in mind, the following weighting factors for each criterion were obtained:

- for accuracy: 16/30 = 8/15 in the case of audio as text, 16/20 = 0.8 in the case of signal;
- for the possibility of taking into account the context: 10/30 = 5/15 in the case of audio as text;
- for saving time of model training: 2/30 = 1/15 in the case of audio as text, 2/20 = 0.1 in the case of audio as a signal;
- for saving the minimum amount of data to achieve the required accuracy: 2/30 = 1/15 in the case of audio as text, 2/20 = 0.1 in the case of audio as a signal.

The next important element of the experimental environment is the determination of possible errors. Based on the described plan, the following factors can be identified that can affect the result:

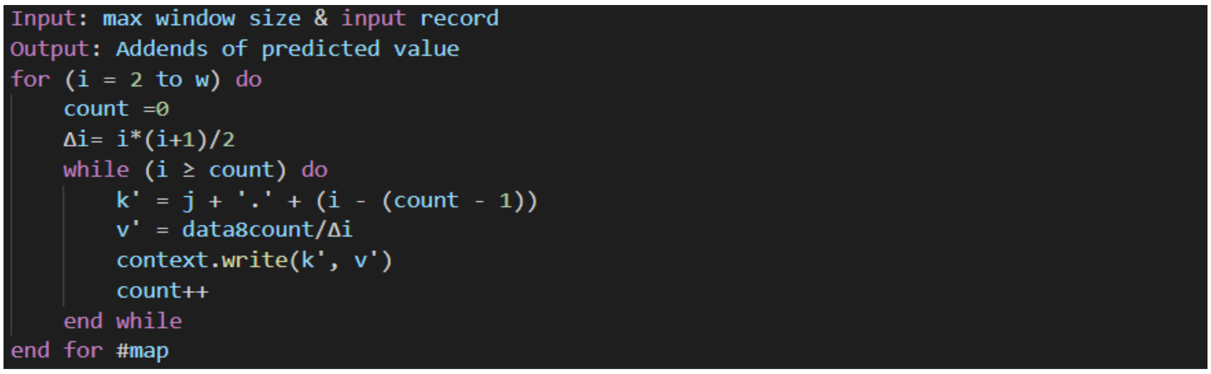
– during time-saving check: human factor and instrumental error;
– during accuracy check: data problem.

To mitigate these uncertainties, as already mentioned, the indicators will be measured several times. Having considered the main aspects of the experimental environment, authors will proceed to the discussion of the implementation of the chosen approach.

5. Models implementation

Since the essence of this process for a signal is relatively trivial, for illustration the augmentation process based on autoregression is considered. As already mentioned, in order to speed up this process, it was decided to use MapReduce. When implementing the selected Hadoop-based technology, the reducer and mapper functions are still the most important, despite the presence of additional structures on the nodes.

In order to make the algorithm language-independent and easier to understand, pseudocode will be provided for each MapReduce function in the future. Mapper shown in Figure 7.



```
Input: max window size & input record
Output: Addends of predicted value
for (i = 2 to w) do
    count =0
    Δi= i*(i+1)/2
    while (i ≥ count) do
        k' = j + '.' + (i - (count - 1))
        v' = data8count/Δi
        context.write(k', v')
        count++
    end while
end for #map
```

Figure 7: Pseudocode for mapping function for basic autoregression

In the case of processing textual information, one of the main stages is to conduct a content analysis, which would allow first to form a dictionary of all important words, and then to find the BM-25 characteristic. As already mentioned, for this it was decided to use the nltk library, which contains a large number of different corpora of words.

The Porter stemmer was chosen for the stemming operation, and the WordNet lemmatizer was used for lemmatization. The specified data structures showed a high accuracy of correct definition of the bases and lemmas of Ukrainian words.

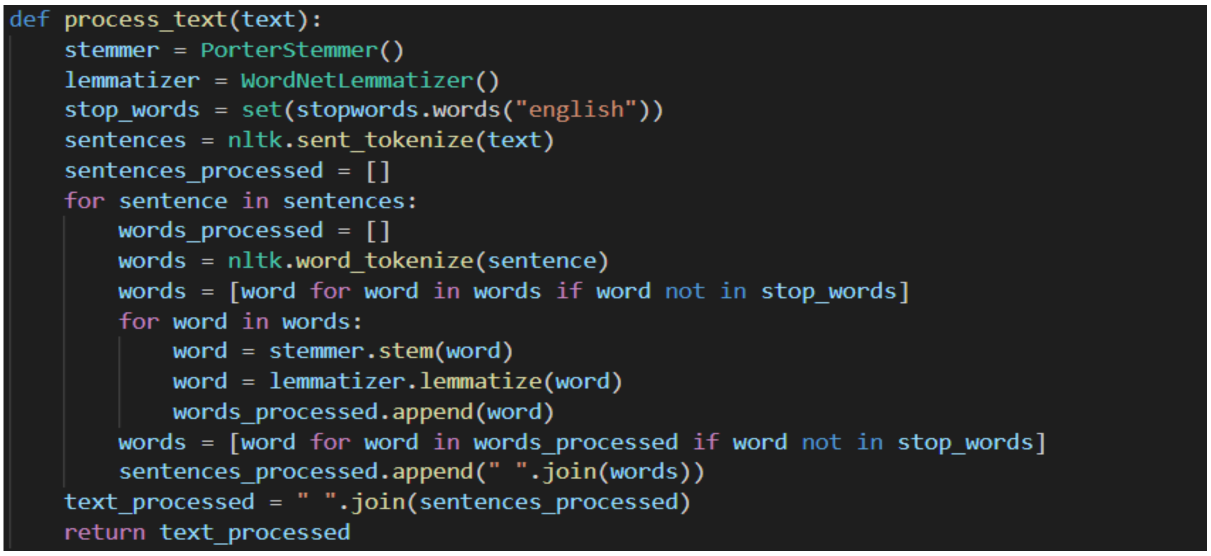
It should be noted that to determine the accuracy, fragments of the results of news text processing were compared with the corresponding processing carried out by linguists from the cities of Kharkiv, Dnipro and Lviv. If dialectics are not done take into account, then the accuracy was 100%. In the case of English, the accuracy score remained unchanged.

The BM-25 components were implemented using the sklearn library. The part of the code that performs dictionary formation for the English language is shown in Figure 8:

After processing, the text data is transferred to the non-relational MongoDB database. The rejection of the classic tabular data storage architecture was due to the following factors:

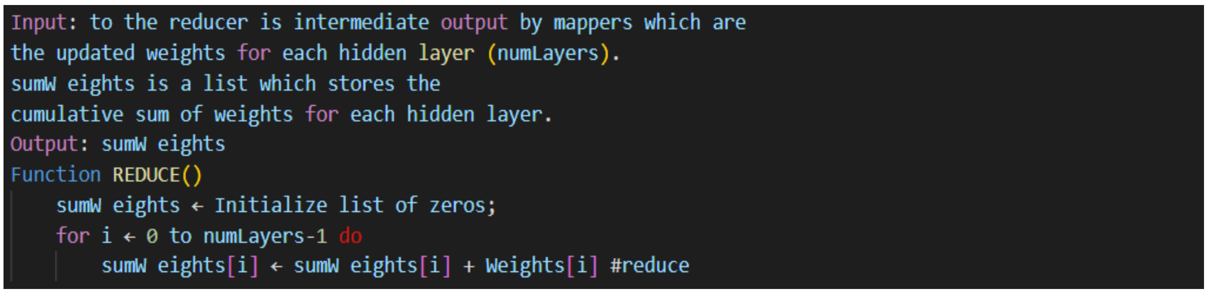
– higher speed of downloading data from the database, which reduced the time of text re-editing;
– the need to save audio files in the database.

The next step in implementation is neural network programming. To avoid implementation errors, as already mentioned, the tensorflow library was used. Although it does not provide an opportunity to create BiLSTM and RCNN architectures directly, it allows the creation of Pipeline structures that can contain several models and reprocessing functions. At the same time, the use of similar structures in combination with our own aggregating structures allows us to build the specified models using MapReduce. For example, autors will give a pseudocode for reduction, which must be used when aggregating the results of several convolutional neural networks. It is shown in Figure 9:



```python
def process_text(text):
    stemmer = PorterStemmer()
    lemmatizer = WordNetLemmatizer()
    stop_words = set(stopwords.words("english"))
    sentences = nltk.sent_tokenize(text)
    sentences_processed = []
    for sentence in sentences:
        words_processed = []
        words = nltk.word_tokenize(sentence)
        words = [word for word in words if word not in stop_words]
        for word in words:
            word = stemmer.stem(word)
            word = lemmatizer.lemmatize(word)
            words_processed.append(word)
        words = [word for word in words_processed if word not in stop_words]
        sentences_processed.append(" ".join(words))
    text_processed = " ".join(sentences_processed)
    return text_processed
```

Figure 8: Python code for dictionary generation



```
Input: to the reducer is intermediate output by mappers which are
the updated weights for each hidden layer (numLayers).
sumW eights is a list which stores the
cumulative sum of weights for each hidden layer.
Output: sumW eights
Function REDUCE()
    sumW eights ← Initialize list of zeros;
    for i ← 0 to numLayers-1 do
        sumW eights[i] ← sumW eights[i] + Weights[i] #reduce
```

Figure 9: Pseudocode for reducing function for CNN

Separately, it is worth noting that the code fragments responsible for communication with the Google Speech to Text API, data transfer to the database and individual elements of reprocessing, as well as the aggregation of results, were decided to be omitted. The decision was made to simplify the statements. Having considered the peculiarities of the implementation of the chosen approach, authors can proceed to the analysis of the results of the conducted experiment.

6. Experiment results

Before starting to present the results, indicate the value of the possibility of taking into account the context for each of the above models:

– CNN – 3 points, the context is taken into account only in the neighborhood in both directions using the convolution function;

– RNN – 2 points, the context is taken into account only in the neighborhood in one direction due to the presence of short-term memory;

– LSTM – 4 points, the context is taken into account only in one direction;

– BiLSTM – 5 points, the context is fully taken into account in both directions due to the bidirectional nature of the network;

– RCNN – 5 points, context is fully taken into account in both directions using convolution functions and a bidirectional network with long-term memory.

Start the overview of experiment results with the data reprocessing time savings indicator for the signal use case (augmented using autoregressive models).

The results are shown in Table 1 (all savings values are calculated relative to the slowest algorithm – the sequential version of integrated moving average vector autoregression).

The following notations were introduced to simplify the statements:

– VAR – simple vector autoregression;

– VARS – seasonal vector autoregression;
– VARL – vector autoregression of the distributed lag;
– VARMA – vector autoregression of the moving average;
– VARIMA – vector autoregression of the integrated moving average.

Table 1
Time-saving of preprocessing audio as signal (in milliseconds)

Sequential					MapReduce				
VAR	VARS	VARL	VARMA	VARIMA	VAR	VARS	VARL	VARMA	VARIMA
55	47	36	12	0	169	138	105	49	4
58	45	30	13	0	173	135	101	52	5
61	48	36	20	0	165	130	99	47	3
57	42	35	22	0	168	132	102	49	4
59	48	38	18	0	166	132	101	50	5

For VAR – 0.058 s, for VARS – 0.046 s, for VARL – 0.035 s, for VARMA – 0.017 s, for VARIMA – 0 s. As you can see, on average, the moving average and integrated moving average algorithms are significantly slower. This is explained by the fact that they take into account exogenous variables in full and at the same time perform noise adjustment. Since in our case, the accuracy of augmentation is not the most important indicator, given the results, it was decided to use classical vector autoregression.

The gain in speed for MapReduce versions is ~ 2.9 for each of the models. At the same time, if you improve the configuration for MapReduce by increasing the number of nodes to 4, then the profit will be ~ 3.74. In the case of audio-to-text analysis, there is only a difference between the parallelized version and the sequential version. In the case of three MapReduce nodes, the acceleration was ~ 3.1, and in the case of four ~ 4.3. The number of nodes is less than the acceleration, due to additional optimization of requests to the Google Speech-to-Text API. The results of the training time-saving measurements and start with the analysis of the audio as a signal show in Table 2. This time, the application of parallelization will be available only for BiLSTM and RCNN architectures, to determine the overall level of speedup. This time, the model built on the RCNN architecture is the slowest.

Table 2
Time-saving of training time for neural networks (in seconds) for audio as signal

Sequential					MapReduce	
CNN	RNN	LSTM	BiLSTM	RCNN	BiLSTM	RCNN
50	48	29	15	0	30	25
49	44	30	14	0	29	24
52	47	28	17	0	31	27
54	45	33	16	0	28	24
50	46	30	18	0	30	26

Authors have the following average values of indicators: CNN: 51 s; RNN: 46 s; LSTM: 30 s; BiLSTM: 16 s; RCNN: 0 s. As a result, the more complex the architecture is, the slower the training of the corresponding model is. In this case, the speedup obtained with MapReduce for BiLSTM was 2 (explained by the ease of parallelization and the limitation to 2 nodes); for RCNN ~ 3.52 (for four nodes the result increased to ~ 4.68). When carrying out similar measurements for training selected neural networks with textual data, the following average time saving results were obtained: CNN: 45 s; RNN: 41 s; LSTM: 24 s; BiLSTM: 12 s; RCNN: 0 s; BiLSTM based MapReduce: 24 s; RCNN based MapReduct: 22 s.

The average time saving is less, which is explained by the specifics of natural language processing. This time, the acceleration obtained using MapReduce for BiLSTM was 2 (the situation is similar to the previous one); for RCNN ~ 3.2 (for 4 nodes the result increased to ~ 4.41). It should be noted that the use of MapReduce did not affect the classification accuracy for both data sets, so the corresponding calculations were omitted. In general, the results are shown in Table 3 for audio as a signal:

Table 3
Classification accuracy (calculated via F1 score and Precision combination) for audio as signal

Dataset	CNN	RNN	LSTM	BiLSTM	RCNN
Election	0.89	0.91	0.93	0.95	0.97
War	0.91	0.89	0.92	0.97	0.97

Considering the obtained result, RCNN guarantees the highest classification accuracy (although the difference with BiLSTM is not significant). In the case of considering audio as text, the situation almost does not change, given the results shown in Table 4.

Table 4
Classification accuracy (calculated via F1 score and Precision combination) for audio as text

Dataset	CNN	RNN	LSTM	BiLSTM	RCNN
Election	0.92	0.91	0.91	0.96	0.96
War	0.90	0.91	0.94	0.95	0.96

The final metric is the training sample size needed to achieve at least 90% accuracy. To do this, several iterations were carried out with a gradual increase in the number of records from 5000 to 10,000 (in the case of audio signals, the vast majority was the result of augmentation).

It is found that the specified accuracy is achieved with 7000 records for CNN; 7500 entries for RNN; 6600 records for LSTM; 6100 entries for BiLSTM; and 5800 entries for RCNN. Thus, the minimum allowable data saving is 1700 for RCNN, 1400 for BiLSTM, 900 for LSTM, 500 for CNN, and 0 for RNN. Now the obtained metric values can be systematized for the case of audio processing as a signal (see Table 5). All unnormalized values were normalized and rounded to the nearest hundredth. Based on the results, the Pareto-optimal alternatives will be determined (see Table 6).

Table 5
Criteria values for each alternative for audio as signal

Model	Time-saving of training time	Accuracy	Min sample size
CNN	1.00	0.90	0.29
RNN	0.90	0.90	0.00
LSTM	0.59	0.93	0.53
BiLSTM	0.31	0.96	0.82
RCNN	0.00	0.97	1.00

Based on the obtained results, it is possible to calculate the value of linear additive convolution with weighting coefficients. For CNN – 0.849, for LSTM – 0.856, for BiLSTM – 0.881, and for RCNN – 0.876. This allows us to note that the most effective model for detecting the fact of falsification for audio as a signal is a bidirectional recurrent neural network with support for long- and short-term memory. However, the difference between BiLSTM and RCNN is not very noticeable and can be considered an error. In addition, the significant speed gain for BiLSTM is partially neutralized by the use of MapReduce technology. Let's proceed to the systematization of the results obtained during the classification of audio as text information. The corresponding normalized values are given in Table 7. Based on the results, the Pareto-optimal alternatives will be determined (see Table 8).

Based on the obtained results, it is possible to calculate the value of linear additive convolution with weighting coefficients. For CNN – 0.771, for LSTM – 0.833, for BiLSTM – 0.918, and for RCNN – 0.917. As in the previous case, the most efficient model is BiLSTM, but the gain in speed is reduced due to parallelization. Considering the above, it can be noted that the most effective models are BiLSTM and RCNN, and the results of the MapReduce application prove the feasibility of its use in the classification of fake audio recordings of various kinds.

Table 6
Criteria values for each Pareto optimal alternative for audio as signal

Model	Time-saving of training time	Accuracy	Min sample size
CNN	1.00	0.90	0.29
LSTM	0.59	0.93	0.53
BiLSTM	0.31	0.96	0.82
RCNN	0.00	0.97	1.00

Table 7
Criteria values for each alternative for audio as text

Model	Time-saving of training time	Accuracy	Min sample size	Context
CNN	1.00	0.91	0.29	0.60
RNN	0.91	0.91	0.00	0.40
LSTM	0.53	0.93	0.53	0.80
BiLSTM	0.27	0.96	0.82	1.00
RCNN	0.00	0.96	1.00	1.00

Table 8
Criteria values for each Pareto optimal alternative for audio as text

Model	Time-saving of training time	Accuracy	Min sample size	Context
CNN	1.00	0.91	0.29	0.60
LSTM	0.53	0.93	0.53	0.80
BiLSTM	0.27	0.96	0.82	1.00
RCNN	0.00	0.97	1.00	1.00

7. Conclusion

The current work aimed to develop an effective model for determining the fact of falsification of sound data, using MapReduce technology. For this purpose, an analysis of the features of falsification of audio information, both in the form of a signal and in textual representation, was carried out. In addition, the analysis of modern scientific publications devoted to the chosen topic and several expert surveys allowed us to form a set of algorithms for creating our model for identifying fake audio. The first stage of this model includes:

– in the case of audio as text: data processing using Google Speech to Text and subsequent conversion of the text into a numerical representation based on: frequency-emotional characteristics (found using the BM-25 algorithm) of the message itself and the latest verified news, degree of conversion reliability, weight message;

– in the case of audio as a signal: cleaning the signal from noise and non-vocalized areas followed by augmentation using vector autoregression parallelized using MapReduce (the results of the experiment allowed us to justify the choice of classic vector autoregression).

The next stage is the application of a neural network. Based on the review of the analyzed studies, it was decided to focus on recurrent and convolutional neural networks. In particular, it is:

– classical convolutional neural network;
– classical recurrent neural network;
– recurrent neural network with long-term memory;
– bidirectional recurrent neural network with long-term memory;

– hybrid neural network combining several convolutional networks with a bidirectional recurrent network with long-term memory.

To overcome the problem of time-related learning of the model, it was decided to use MapReduce technology. To determine the most effective neural network and the expediency of using the proposed parallelization method, a set of criteria was formed that allowed using the principle of linear additive convolution with weighting coefficients. Based on these criteria, specified modifications and implementation of selected models using Python 3 libraries, a series of experiments was conducted with data on the election process in Ukraine in 2019 and the full-scale invasion of the Russian Federation on the territory of Ukraine.

In the course of the experiments, it was found that the bidirectional recurrent neural network with long-term memory is the most effective, although it loses in speed to less complex models. At the same time, the difference in efficiency between it and the hybrid neural network is insignificant. It is found that the gain in reprocessing time saving when using MapReduce technology can reach 4.3 in the case of text and 4 in the case of signal. At the same time, the gain in neural network training time savings can reach 4.71 in the case of text and 4.68 in the case of signal, bridging the gap in performance between BiLSTM and RCNN. Therefore, authors can claim that the use of a built model based on BiLSTM (or RCNN) is highly effective for determining the fact of audio forgery both in the form of text and the form of a signal. At the same time, the implementation of the process of data processing and training of neural networks using MapReduce is appropriate. Thus, the goal of this study has been fulfilled.

8. Acknowledgements

The authors would like to thank the Armed Forces of Ukraine for the opportunity to write a valid work during the full-scale invasion of the Russian Federation on the territory of Ukraine.

9. References

[1] Anders, M., "Fake News Detection", *European Data Protection Supervisor*, available at: https://edps.europa.eu/press-publications/publications/techsonar/fake-news-detection_en (last accessed 14.10.2023).

[2] Bansal, N., Aljrees, T., Yadav, D. P., Singh, K. U., Kumar, A., Verma, G. K., & Singh, T. (2023), "Real-Time Advanced Computational Intelligence for Deep Fake Video Detection", *Applied Science*, No. 13(5), Article 3095. DOI: 10.3390/app13053095.

[3] Batailler, C., Brannon, S. M., Teas, P. E., & Gawronski, B. (2023), "A Signal Detection Approach to Understanding the Identification of Fake News", *Perspectives on Psychological Science*, No. 17(1), P. 78–98. DOI: 10.1177/1745691620986135.

[4] Reis, J. C. S., Correia, A., Murai, F., Veloso, A., & Benevenuto, F. (2019), "Supervised Learning for Fake News Detection", *IEEE Intelligent Systems*, No. 34(2), P. 76–81. DOI: 10.1109/MIS.2019.2899143.

[5] Giandomenico, D. D., Sit, J., Ishizaka, A., & Nunan, D. (2021), "Fake news, social media and marketing: A systematic review", *Journal of Business Research*, Vol. 124, P. 329–341. DOI: 10.1016/j.jbusres.2020.11.037.

[6] Yuan, L., Jiang, H., Shen, H., Shi, L., & Cheng, N. (2023), "Sustainable Development of Information Dissemination: A Review of Current Fake News Detection Research and Practice", *Systems*, No. 11(9), Article 458. DOI: 10.3390/systems11090458.

[7] Rocha, Y. M., de Moura, G. A., Desiderio, G. A., de Oliveira, C. H., Lourenço, F. D., & de Figueiredo Nicolete, L. D. (2023), "The impact of fake news on social media and its influence on health during the COVID-19 pandemic: a systematic review", *Journal of Public Health*, Vol. 31, P. 1007–1016. DOI: 10.1007/s10389-021-01658-z.

[8] Bondielli, A., Dell'Oglio, P., Lenci, A., Marcelloni, F., Passaro L. C., & Sabbatini, M. (2023), " MULTI-Fake-DetectiVE at EVALITA 2023: Overview of the MULTImodal Fake News Detection and VErification Task". *Evaluation Campaign of Natural Language Processing and Speech Tools for Italian, (EVALITA 2023): 8th Internaional Conference, Parma, 7 September – 8 September 2023: CEUR Workshop Proceedings.* No. 3373, available at: https://ceur-ws.org/Vol-3473/paper32.pdf (last accessed: 14.10.2023).

[9] Sardar, T. H., & Ansari, Z. (2020), “An Analysis of Distributed Document Clustering Using MapReduce Based K-Means Algorithm”, *Journal of The Institution of Engineers (India): Series B,* Vol. 101, P. 641–650. DOI: 10.1007/s40031-020-00485-2z.

[10] Deng, R., & Duzhin, F. (2022), “Topological Data Analysis Helps to Improve Accuracy of Deep Learning Models for Fake News Detection Trained on Very Small Training Sets”, *Big Data and Cognitive Computing,* No. 6(3), Article 74. DOI: 10.3390/bdcc6030074.

[11] Choudhary, A., & Arora, A. (2021), “Linguistic feature based learning model for fake news detection and classification”, *Expert Systems with Applications,* No. 169, Article 114171. DOI: 10.1016/j.eswa.2020.114171.

[12] Alonso, M. A., Vilares, D., Gómez-Rodríguez, C., & Vilares, J. (2021), “Sentiment Analysis for Fake News Detection”, *Electronics,* No. 10(11), Article 1348. DOI: 10.3390/electronics10111348.

[13] Tolosana, R., Vera-Rodriguez, R., Fierrez, J., Morales, A., & Ortega-Garcia, J. (2020), “Deepfakes and beyond: A Survey of face manipulation and fake detection”, *Information Fusion,* Vol. 64, P. 131–148. DOI: 10.1016/j.inffus.2020.06.014.

[14] Afanasieva, I., Golian, N., Golian, V., Khovrat, A., & Onyshchenko, K. (2023), "Application of Neural Networks to Identify of Fake News". *Computational Linguistics and Intelligent Systems (COLINS 2023): 7th International Conference, Kharkiv, 20 April – 21 April 2023: CEUR workshop proceedings,* No. 3396, P. 346–358, available at: https://ceur-ws.org/Vol-3396/paper28.pdf (last accessed: 14.10.2023).

[15] Bhatia, N. (2020), “Using transfer learning, spectrogram audio classification, and MIT app inventor to facilitate machine learning understanding”, *Massachusetts Institute of Technology,* available at: https://dspace.mit.edu/handle/1721.1/127379 (last accessed: 14.10.2023).

[16] Breuer, A., Eilat, R., & Weinsberg, U. (2020), " Friend or Faux: Graph-Based Early Detection of Fake Accounts on Social Networks". *Web Conference (WWW ‘20): International Conference, Taipei, 20 April – 24 April 2023: Association for Computing Machinery,* P. 1287–1297. DOI: 10.1145/3366423.3380204.

[17] Xia, T., & Chen, X. (2020), “A Discrete Hidden Markov Model for SMS Spam Detection”, *Applied Science,* No. 10(14), Article 5011. DOI: 10.3390/app10145011.

[18] Najar, F., Zamzami, N., & Bouguila, S. (2019), "Fake News Detection Using Bayesian Inference". *Information Reuse and Integration for Data Science (IRI 2019): 20th IEEE International Conference, Los Angeles, 30 July – 1 August 2019: IEEE,* P. 389–394. DOI: 10.1109/IRI.2019.00066.

[19] Montserrat, D. M., Horváth, J., Yarlagadda, S. K., Zhu, F., & Delp, E. J. (2020), " Generative Autoregressive Ensembles for Satellite Imagery Manipulation Detection". *Workshop on Information Forensics and Security (WIFS 2020): 12th IEEE International Workshop, New York, 6 December – 11 December 2020: IEEE,* P. 1–6. DOI: 10.1109/WIFS49906.2020.9360909.

[20] Ning C., & You F. (2019), “Optimization under uncertainty in the era of big data and deep learning: When machine learning meets mathematical programming”. *Computers & Chemical Engineering,* Vol. 125, P. 434–448. DOI: 10.1016/j.compchemeng.2019.03.034.

[21] Khovrat, A., Kobziev, V., Nazarov, A., & Yakovlev, S. (2022), "Parallelization of the VAR Algorithm Family to Increase the Efficiency of Forecasting Market Indicators During Social Disaster". *Information Technology and Implementation (IT&I 2022): 9th Internaional Conference, Kyiv, 30 November – 2 December 2022: CEUR Workshop Proceedings.* No. 3347, P. 222–233, available at: https://ceur-ws.org/Vol-3347/Paper_19.pdf (last accessed: 14.10.2023).

[22] "Long Short-Term Memory Networks (LSTM)", *Data Base Camp,* available at: https://databasecamp.de/en/ml/lstms (last accessed 14.10.2023).

[23] Zvornicanin, E., "Differences Between Bidirectional and Unidirectional LSTM", *Baeldung,* available at: https://www.baeldung.com/cs/bidirectional-vs-unidirectional-lstm (last accessed 14.10.2023).

[24] Ashley, "An Overview on Convolutional Neural Networks", *Medium,* available at: https://medium.com/swlh/an-overview-on-convolutional-neural-networks-ea48e76fb186 (last accessed 14.10.2023).

[25] Cho, D., Lee, H., Lee, W., & Kang, S. (2021), “Detecting Anomalous Kicks in Taekwondo With Spatial and Temporal Features”, *IEEE Access,* Vol. 9, P. 164928–164934. DOI: 10.1109/ACCESS.2021.3134967.

[26] Al-Khasawneh M. A., Uddin, I., Shah, S. A. A., Khasawneh, A. M., Abualigah, L. & Mahmoud, M. (2022), “An improved chaotic image encryption algorithm using Hadoop-based MapReduce framework for massive remote sensed images in parallel IoT applications”. *Cluster Computing,* Vol. 25, P. 999–1013. DOI: 10.1007/s10586-021-03466-2.