INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY RESEARCH AND ANALYSIS

ISSN(print): 2643-9840, ISSN(online): 2643-9875

Volume 06 Issue 02 February 2023

DOI: 10.47191/ijmra/v6-i2-20, Impact Factor: 6.261

Page No. 627-646

Research, Direct Construction form Optical Characteristics – OCR using Department Technology

Ly Hai Son[1], Nguyen Thu Nguyet Minh[2], Tra Van Dong[3]

[1]Faculty of Fundamental Science, Van Lang University, 69/68 Dang Thuy Tram Street, Ward 13, Binh Thanh District, Ho Chi Minh City, Vietnam.

[2]Faculty of Fundamental Science, Van Lang University, 69/68 Dang Thuy Tram Street, Ward 13, Binh Thanh District, Ho Chi Minh City, Vietnam. ORCID ID: 10000-0001-5103-2553

[3]Faculty of Fundamental Science, Van Lang University, 69/68 Dang Thuy Tram Street, Ward 13, Binh Thanh District, Ho Chi Minh City, Vietnam.

ABSTRACT: Optical character recognition – using a deep learning technique that describes an overview of optical character recognition, the basic steps in the optical character recognition problem. At the same time, the thesis introduces in detail the Long short term memory (LSTM) deep learning method and its application in optical character recognition. Through testing on a dataset of 10,000 license plate photos, the thesis showed that the application of the LSTM method is quite effective in optical character recognition.

KEYWORDS: optical character, optical character recognition

1. INTRODUCTION

Optical Character Recognition is computer software created to convert handwritten or typed images into document documents.

Optical character recognition is formed from the fields of study of pattern recognition, artificial intelligence, and computer vision. Although academic research work continues, part of OCR's work has shifted to practical application with proven techniques. Nowadays, optical character recognition techniques have been widely used and applied in practice in parallel with theoretical research to improve recognition results.

Typically, optical character recognition systems are used in the form of software in computers or integrated in printers and scanners to perform character recognition. The most common example is scanning text images into documents stored on a computer.

Currently, optical character recognition solutions are thriving in terms of application and continuously have many new improvements to increase the applicability and efficiency of the output product. At the forefront of OCR technology are two companies that develop and improve character recognition software, Google and ABBYY.

+ Google on the Tesseract platform (Tesseract OCR engine) developed by HP Labs in the period 1985-1995, using open source, high accuracy recognition quality, with many image file formats and can recognize more than 60 different languages.

+ Meanwhile, ABBYY is considered a pioneer in the field of OCR. ABBYY released optical character recognition software called ABBYY capable of recognizing 190 languages. In particular, for Latin and Russian characters, ABBYY's OCR technology can achieve up to 99% recognition efficiency for a good quality image file.

+ However, the recognition of Vietnamese characters (the type of language with "accents") is still a challenge for the development of OCR technology in the world. Currently, ABBYY is conducting research and deploying Vietnamese recognition technology with over 90% accuracy for a good quality image file. However, it has only stopped at recognizing Vietnamese characters drafted by computers or the printing industry without being able to reach the types of handwriting, even the efficiency achieved is very modest for old documents, diverse fonts or using outdated printing techniques. In addition, ABBYY's OCR solution lacks competitiveness because of its high price, foreign packaged products, not really suitable for Vietnamese documents, as well as difficulties in integrating into other technologies.

- In Vietnam, there are also some software companies investing in building OCR technology, such as the following:

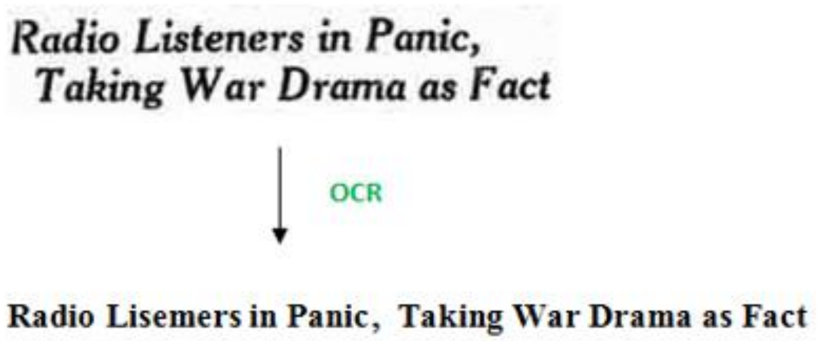
+ VnDOCR 4.0 Professional software, a printed Vietnamese letter recognition program, developed by a team of software development experts of the Department of Identification and Knowledge Technology, Institute of Information Technology - Vietnam Institute of Science and Technology. VnDOCR uses a scanner control program, to scan images from printed documents as line art, Black and White - B&W, with a resolution of 300dpi (dots per inch), then switch to recognition mode. Vietnamese word recognition results are about 90% accurate depending on the quality of the scan.

+ VietOCR software is developed based on the open source Tesseract platform, with Java / .NET technology, supporting recognition for PDF, TIFF, JPEG, GIF, PNG, and BMP image formats. VietOCR's recognition ability can reach 95% for good quality image files.

2. LITERATURE REVIEW HISTORY OPTICAL CHARACTER RECOGNITION PROCESS

First, the recognition system requires training with specific character patterns. "Smart" systems with high recognition accuracy for most fonts are now commonplace. Some systems are also capable of reproducing the formatting of documents that closely resemble the original, including: images, columns, tables, non-text elements.

The input of this process is the image file and the output will be the text file containing the text text contained in that image.



Radio Listeners in Panic,
Taking War Drama as Fact

Radio Lisemers in Panic, Taking War Drama as Fact

Figure 2. 1: OCR implementation process

Building an OCR tool is a step-by-step process. The development process usually consists of 6 steps necessary to train an algorithm to solve the problem effectively with the help of optical character recognition:

+ Image acquisition: The first step is to collect images of paper documents with the help of optical scanners. In this way, an original image can be captured and stored. Most paper documents are black and white and OCR scanners will be able to threshold images. In other words, it should replace each pixel in the photo with a black or white pixel. It's an image segmentation method.

+ Preprocessing: makes the raw data usable by computer. The level of noise on the image must be optimized and areas outside the text removed. Preprocessing is especially important for recognizing handwritten documents that are more sensitive to noise. Preprocessing allows to obtain a clean character image to deliver better image recognition results.

+ Segmentation: aims to group characters into meaningful blocks. There may be predefined classes for the characters. So images can be scanned for patterns that match the layers.

+ Feature mining: to divide the input data into a set of features, that is, to find essential characteristics that make one or another pattern recognizable. As a result, each character is classified in a specific class.

+ Train a neural network: Once all features are extracted, they can be fetched into the neural network (NN) to train it to recognize characters. A training dataset and the methods adopted to achieve the best output will depend on a problem that requires an OCR-based solution.

+ Post-processing: the screening process because an OCR model may require some modifications. However, 100% accuracy cannot be achieved. The identification of the characters depends a lot on the setting. Verifying the output requires a human loop approach.

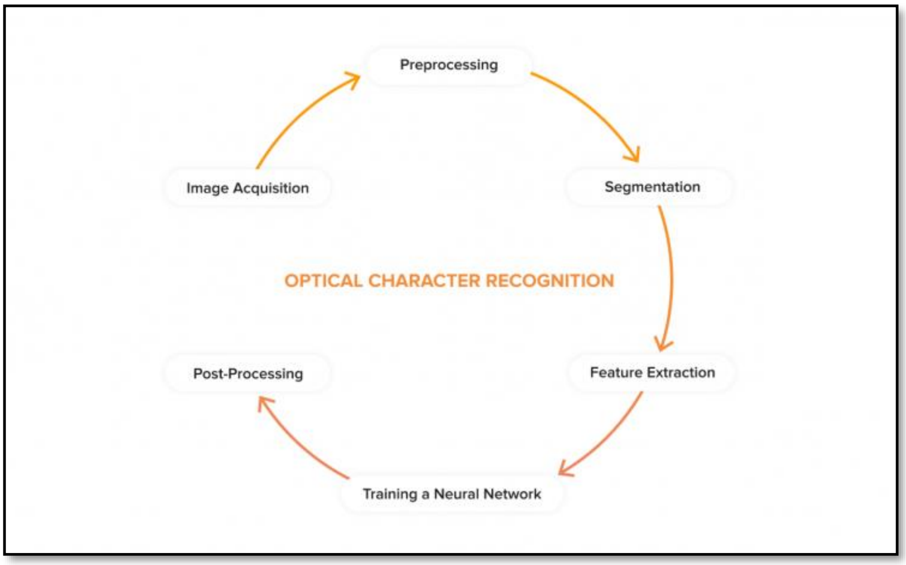


Figure 2. 2: Steps to build OCR master.

2.1. The main problems of optical character recognition:

Overall, OCR technology has created a new, groundbreaking technical solution for building electronic databases. However, current software only achieves high efficiency for good quality text files, along with some types of Latin languages.

The correct identification of typed Latin characters is considered a problem that has been solved. The actual accuracy rate reaches 99%, although some applications require even higher accuracy rates that require human error checking.

Hand-printed letter recognition, hand-written cursive, and even typed versions of some letters (especially those with large numbers of letters), remain a subject of study.

Handwritten character recognition systems have enjoyed great commercial success in recent years. Among them are imported devices for personal assistance devices (PDAs) such as those running on Palm OS and Apple Newton's pioneering technology. The algorithms used in these devices use the advantage that the order, speed, and direction of single lines are already known. Similarly, users may be required to use only certain types of typefaces.

These methods cannot be used in paper document scanning software, so accurate recognition of hand-printed text remains a major issue. With an accuracy of 80% to 90%, clean hand-printed characters can be recognized, but that accuracy still produces dozens of errors per page, making that technology only effective in certain cases. The variety of OCR is now known in the industry as ICR, (Intelligent Character Recognition).

Handwriting recognition is a vibrant field of study, with recognition rates even lower than hand-printed text. Higher recognition rates of generic manuscripts are virtually impossible without the use of grammatical and contextual information. For example, it's easier to identify an entire word from a dictionary than it is to try to extract discrete characters from that paragraph. Reading the total of a check (always written in numbers) is an example where using smaller dictionaries can greatly increase recognition rates. Knowledge of the grammar of a scanned language can also help identify whether a word may be a verb or noun, for example, which will allow for a more accurate 2/3 Optical Character Recognition (OCR). The shape of the handwriting itself did not contain enough information about it to accurately identify (more than 98%) all the handwriting fragments.

In short, identity problems that are more complex than neural networks are widely used because they can simplify both affine and nonlinear transformations.

3. DESCRIPTION OF THE PROBLEM

Currently, the need to extract words from images is growing. Convert images of handwriting or typing into characters that have been encoded in the computer. Let's say we need to edit some paper documents such as magazine articles, flyers, or an image PDF file. Obviously, we cannot use a scanner to convert these documents into text files that can be edited (e.g. Microsoft Word editor).

3.1. Traditional methods

In addition to manual methods, there are now quite a few sets of optical character recognition libraries in the world with quite high accuracy. Using one of those libraries will save us quite a lot of effort. The following are some of the free optical character recognition suites and software that are widely used today:

+ Tesseract OCR: is a commercial optical character recognition unit originally developed at HP (Hewlett-Packard) between 1985 and 1995 and was awarded the top 3 most accurate optical character recognition software at the annual conference of UNLV (University of Nevada-Las Vegas). This identifier was later turned into open source on Google and continues to be developed to this day with the contributions of many professional programmers. The current head of the project is Ray Smith.

+ GOCR: Is an optical character recognition program developed under the GNU General Public License and started by Joerg Schulenberg in 2000.

+ FreeOCR: Considered one of the most accurate optical character recognition software because it uses HP's Tesseract engine.

+ JavaOCR: Is an optical character recognition software written entirely in Java library for image processing and character recognition. The advantage of this program is that it takes up little memory resources, is easy to implement on memory-limited mobile environments, and can only use the Java language.

- Among the above optical character recognition libraries, the Tesseract OCR is the most outstanding with the following advantages:

+ Have a long development history and bring high accuracy right from the launch.

+ Highly scalable and customizable and sponsored by Google and a large number of developers contributing to Tesseract.

+ The version is updated regularly, supports more and more languages, has the ability to train on new languages and many different types of fonts.

+ Some OCR software now uses this identifier for character recognition, so Tesseract has become more popular, and also has the ability to support on many different environments and platforms from computers to mobile devices.

3.2. Limitations and shortcomings of traditional methods

- According to a report by The Intersect Group, 60% of financial costs are tracked for labor such as manual data entry. Consider the data entry time for your business now and how much it would be worth if your organization eliminated that step.
- It takes time to manually type 1 physical text into text on an editable machine.
- It takes a lot of manpower and time to dissect form information, many errors.

3.3. Artificial intelligence, deep learning techniques

*** Artificial Intelligence**

- Introduction to artificial intelligence

+ Artificial Intelligence is a discipline in the field of Computer Science. It is human-programmed intelligence with the goal of helping computers automate intelligent behaviors like humans.

+ Artificial intelligence differs from logical programming in programming languages in the application of machine learning systems (Machine learning) to simulate human intelligence in processes that humans do better than computers. Specifically, artificial intelligence helps computers acquire human intelligence such as: Knowing how to think and reason to solve problems, know how to communicate by understanding language, speech, learning and self-adaptation.

- *The evolution and development of Artificial Intelligence:*

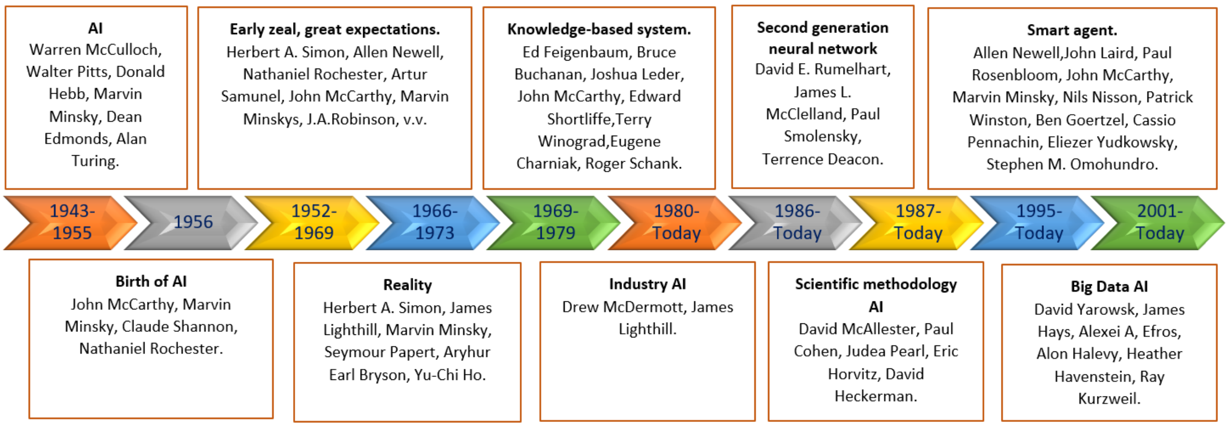

AI
Warren McCulloch, Walter Pitts, Donald Hebb, Marvin Minsky, Dean Edmonds, Alan Turing.
Early zeal, great expectations.
Herbert A. Simon, Allen Newell, Nathaniel Rochester, Artur Samunel, John McCarthy, Marvin Minskys, J.A.Robinson, v.v.
Knowledge-based system.
Ed Feigenbaum, Bruce Buchanan, Joshua Leder, John McCarthy, Edward Shortliffe,Terry Winograd,Eugene Charniak, Roger Schank.
Second generation neural network
David E. Rumelhart, James L. McClelland, Paul Smolensky, Terrence Deacon.
Smart agent.
Allen Newell,John Laird, Paul Rosenbloom, John McCarthy, Marvin Minsky, Nils Nisson, Patrick Winston, Ben Goertzel, Cassio Pennachin, Eliezer Yudkowsky, Stephen M. Omohundro.
1943-1955
1956
1952-1969
1966-1973
1969-1979
1980-Today
1986-Today
1987-Today
1995-Today
2001-Today
Birth of AI
John McCarthy, Marvin Minsky, Claude Shannon, Nathaniel Rochester.
Reality
Herbert A. Simon, James Lighthill, Marvin Minsky, Seymour Papert, Aryhur Earl Bryson, Yu-Chi Ho.
Industry AI
Drew McDermott, James Lighthill.
Scientific methodology AI
David McAllester, Paul Cohen, Judea Pearl, Eric Horvitz, David Heckerman.
Big Data AI
David Yarowsk, James Hays, Alexei A, Efros, Alon Halevy, Heather Havenstein, Ray Kurzweil.

Figure 3. 1: Summary of the evolution of Artificial Intelligence In each stage there is a list of typical artificial intelligence scientists.

Figure 3. 1 Summary of the evolution of artificial intelligence through ten stages from 1943 to the present, synthesized by S. Russell and P. Norvig. The expansion of artificial intelligence, which goes too far from its original origins, also made some artificial intelligence founders (John McCarthy, Marvin Minsky, ...) disgruntled, as they argue that artificial intelligence needs to focus on the original goal of creating "machines that think, learn and create". However, practice has demonstrated that this expansion, especially artificial intelligence with big data, has created exponential development artificial intelligence technologies and platforms in the current period.

S. Russell and P. Norvig argue that artificial intelligence has gone through cycles of success, which can lead to over-optimism leading to a decline in enthusiasm and funding, but at the same time, there are cycles with new creative approaches, to achieve greater achievements. S. Russell and P. Norvig list current artificial intelligence topics as self-driving cars, speech recognition, autonomous planning and scheduling, game consoles, anti-garbage, logistics planning, robotics, machine translation.

The evolution of artificial intelligence indicates that the achievement of each of the following stages is the result of inheritance, promotion of suitable parts and the reduction and correction of inappropriate parts from previous stages. One aspect of artificial intelligence has a qualitative change, aware that such a change is the result of a process of quantitative change.

- Key Areas of Artificial Intelligence:

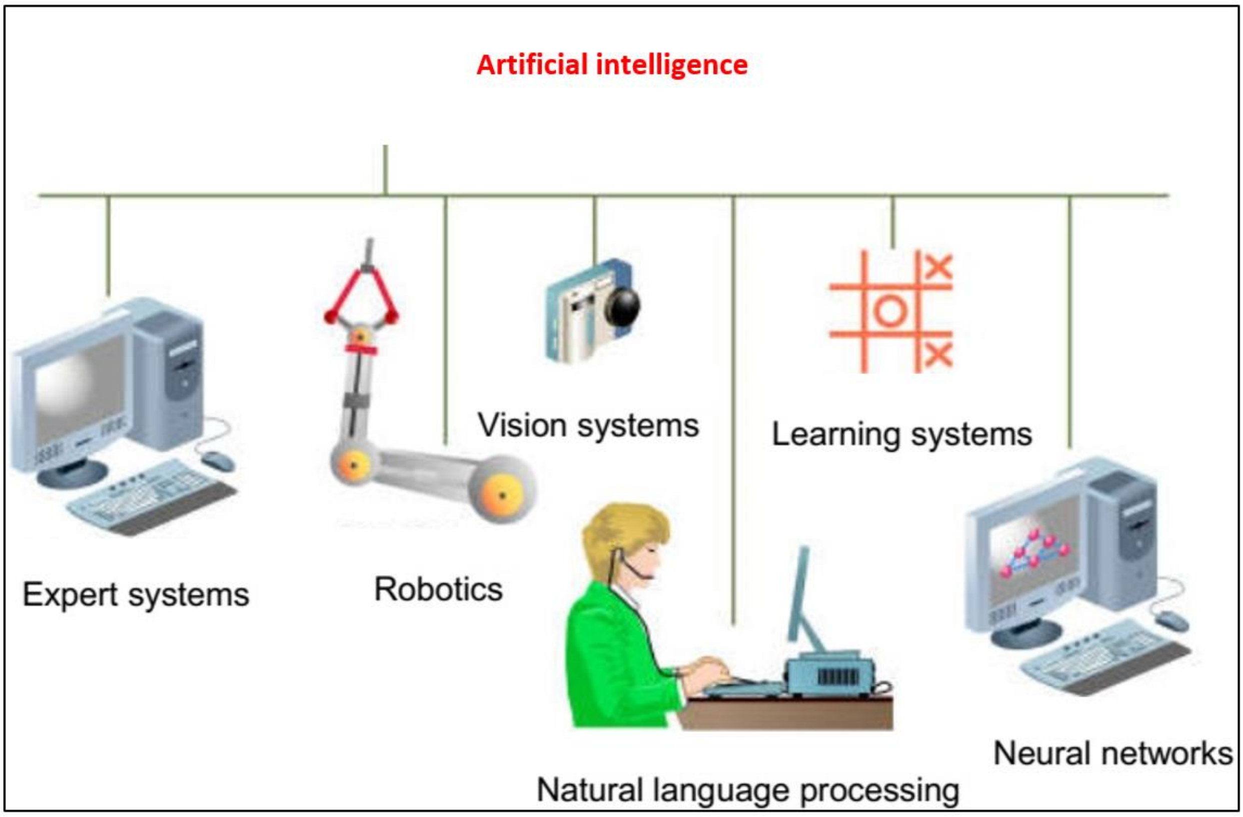


Figure 3. 2: Areas of Artificial Intelligence

Figure 3.2 shows the main areas of artificial intelligence as expert systems, robotics, machine vision systems, natural language processing systems, learning systems, and neural networks.

- The expert system handles consulting situations (identifying consulting problems, collecting data information, inferring problem solving, selecting appropriate solutions), similar to human experts in specific application domains.

- Artificial intelligence robots can perform behaviors with human-like intelligence, thanks to being equipped with software systems and artificial intelligence devices . To minimize the risks in the exploitation and use of artificial intelligence robots, Three laws of robot operation need to be followed:

+ Robots do not take actions that harm humans and need to act accordingly when humans are harmed;

+ Robots obey human commands, except for orders that harm humans (so as not to conflict with the first operational law);

+ Robots know how to protect themselves except in cases of conflict with the first operating law and the second operation law. It is necessary to distinguish artificial intelligence robots from industrial robots doing dull, toxic and dangerous jobs.

- Machine vision systems are capable of recognizing from images: objects, events, processes in the surrounding real world environment and establishing the location of these objects. The machine vision system has the following functions:

+ Object recognition;

+ Locating objects in space;
+ Sticking, navigating, tracking moving objects;
+Acknowledge the behavior of the subject.

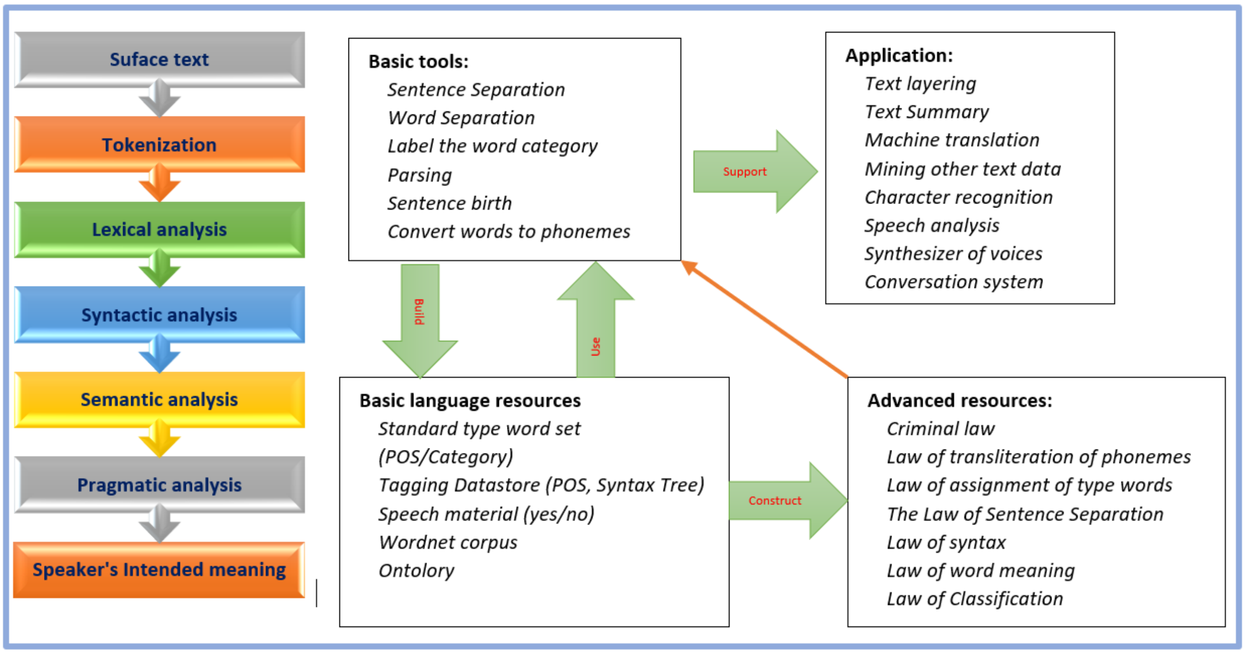


Figure 3. 3left), language tools and resources in natural language processing (right)

Natural language processing systems (Natural language processing, computational linguistics, human language technology, computer speech and language processing) make computers capable of understanding and reacting when receiving sentences and directives expressed in natural languages such as Vietnamese, English... Natural language processing is an artificial intelligence research area that has had a long development process of seven decades, attracting a large research community around the world and also in Vietnam. Natural language processing includes word processing, speech processing, and speech-text processing. Figure 3 gives a view of language tools and resources and their relationships in natural language processing.

- Human knowledge is obtained from three sources:

+ Biological continuation: through the evolution of human survival inherited through generations;

+ Cultural acquisition: acquired through language used by parents, families and teachers to pass on knowledge to the next generation;

+ Lifelong self-education: accumulation of knowledge and skills by individuals. Lifelong self-study helps people upgrade their learning capacity to learn faster and more effectively. Machine learning in artificial intelligence towards computers has the same "learning" (knowledge acquisition) capacity as humans, thanks to knowledge that improves the way it works, responding when receiving feedback from the external environment in situations. Statistical machine learning, especially deep learning, along with big data, is currently a key trend, creating the miraculous development of artificial intelligence for more than a decade. Transfer learning, deep transfer learning, lifelong machine learning are modern machine learning techniques that allow solving problems in situations where critical information is missing or handling new situations.

- Neural networks are areas of artificial intelligence that allow simulated computer systems to act like the human brain in learning data patterns and guessing input layers. Neural network systems typically use a parallel architecture of array processors based on a network structure similar to the human brain.

-Typical problems applying artificial intelligence methods

+ Pattern recognition

- *Optical character recognition*
- Handwriting recognition
- Speech recognition
- Get a face

+ Natural language processing, Automatic translation (machine translation) and Chatterbot

+ Nonlinear Control and Robotics

+ Computer vision, Virtual Reality and Image Processing

+ Game theory and *Strategic planning*
+ Artificial intelligence games and Computer game bots

*** Deep learning**

Deep learning (also known as deep structured learning or hierarchical learning) is part of a broader group of machine learning methods based on representations of learning data, as opposed to task-specific algorithms. Learning can be supervised, semi-supervised or unsupervised.

+ Deep learning architectures such as deep neural networks, deep trust networks and recurrent neural networks have been applied to areas including computer vision, speech recognition, natural language processing, sound recognition, social network filtering, machine translation , bioinformatics, drug design, medical image analysis, material testing, and board game shows, where they have produced results comparable to and in some cases superior to human experts.

+ Deep learning is a class of machine learning algorithms that:

Use a multi-layered cascade of nonlinear processing units to extract characteristics and convert. Each successive class uses the output from the previous class as input. These algorithms can be monitored or unattended, and applications include analytical (unsupervised) and classification (monitoring) models.

Based on learning (without supervision) of multiple levels of characteristics or representations of data. High-end features originate from lower-level features to form a hierarchical representation.

As part of the broader field of machine learning of data representation.

Learn multiple levels of representation corresponding to different levels of abstraction; the levels form a hierarchy of concepts.

- **History**

Deep learning architectures, especially those built from artificial neural networks (ANN), dominated at least until Neocognitron introduced by Masahiko Fukushima in 1980. It was the ANNs who dominated for even longer. The challenge is how to train this network with multiple layers. In 1989, Yann Le Cun and colleagues were able to apply standard reverse transmission algorithms, dating back to 1974, to a deep neural network for the purpose of recognizingZIP code handwriting in letters. Despite the success in applying this algorithm, the time to train the network based on this metric takes about 3 days, making it impractical to use it for normal purposes. In 1995, Brendan Frey demonstrated that it was possible to train a neural network consisting of a full six layers of connectivity and several hundred hidden units using a sleep-wake algorithm, which was developed in collaboration with Peter Dayan and Geoffrey Hinton. However, the training took two days.

Many factors contribute to the reason for the slow pace, one being the gradient disappearance problem analyzed in 1991 by Sepp Hochreiter.

In 1991 such neural networks were used to recognize isolated 2-D handwritten digits, 3-D object recognition was performed by combining 2-D images with a manual 3-D object model. Juyang Weng and colleagues proposed that a human brain does not use a monolithic 3-D object model, and in 1992 they published Cresceptron, a method for performing 3-D object recognition directly from cluttered backscenes. Cresceptron is a stratigraphic composite of layers similar to Neocognitron. But while Neocognitron requires a human programmer to intervene, Cresceptron automatically learns some unsupervised characteristics in each class, where each trait is represented by a convolutional multiplier. Cresceptron also segments each object learned from a messy background scene through the reverse analysis of that network. The max probe, now commonly adopted by deep neural networks (e.g., the ImageNet test), was first used in Cresceptron to reduce the position resolution by a factor (2x2) to 1 through better generalized cascading. Despite these advantages, simpler models using specific tasks with manual characteristics such as Gabor and SVM-support vector machines were popular choices in the 1990s and 2000s, because of the costs calculated by ANNs and because of a lack of understanding of how brains self-manage outcomes. its biological networking.

In the long history of speech recognition, both agrolearning and deep learning (e.g., recurrent networks) of artificial neural networks have been explored for many years. But these methods never won over the manual-internal Gaussian hidden markov modeling/mixed modeling (GMM-HMM) technology based on biophysical models of explicit speech recognition training. Some of the main difficulties have been methodically analyzed, including gradient reduction and weak time correlation structures and in neural predictive models. Additional difficulties were a lack of large training data and weak computing power in the initial period. So most speech recognition researchers who already understood such barriers moved away from neural networks to pursue a physical model, until a recent resurgence of deep learning overcame all these difficulties. Hinton and his colleagues and Dang and colleagues looked at part of this history in terms of how they collaborated with each other and then with colleagues between groups relaunched neuron network research and began deep learning research and speech recognition applications.

- **Conclude**

Many of today's most advanced machine learning systems use neural networks to process data. Recent successes in the driverless car industry have been made possible by deep learning, while principles are also being implemented in the defense and aerospace sectors to identify objects from space, optical character recognition is also a very hot issue of deep learning.

While the potential of deep learning is huge, there are still limitations when it comes to performing human-like multitasking. Deep learning excels at pattern recognition, much like Go's complex but fixed rules. But the researchers point out that the vast amount of data needed to teach a machine is just a specific set of rules.

At the current stage of development, it is not yet possible to develop deep learning to perform complex, adaptive human thought processes, but this technology is still developing at a fairly high rate.

4. APPLICATION OF DEEP LEARNING TECHNIQUES IN OPTICAL CHARACTER RECOGNITION

4.1. Long Short Term Memory Method:

- Long Short Term Memory networks, commonly known as LSTMs, are a special form of RNN, capable of learning distant dependencies.
- LSTM is designed to avoid the problem of long-term dependency. Remembering information for long periods of time is their default feature, but we don't need to train it to remember it. That is, its very essence can be remembered without any intervention.
- LSTM - is a special form of RNN: The main idea of RNN (Recurrent Neural Network) is to use strings of information. In traditional neural networks all inputs and outputs are independent of each other. That is, they are not linked in chains with each other. But these models are not suitable in a lot of problems. For example, if we want to guess how the next word may appear in a sentence, we also need to know how the previous words appear one after the other, right? RNNs are called recurrents because they perform the same task for all elements of a sequence whose output depends on previous calculations. In other words, the RNN is capable of remembering previously calculated information. In theory, the RNN can use the information of a very long text, but in reality it can only remember a few previous steps. Basically an RNN network looks like this:

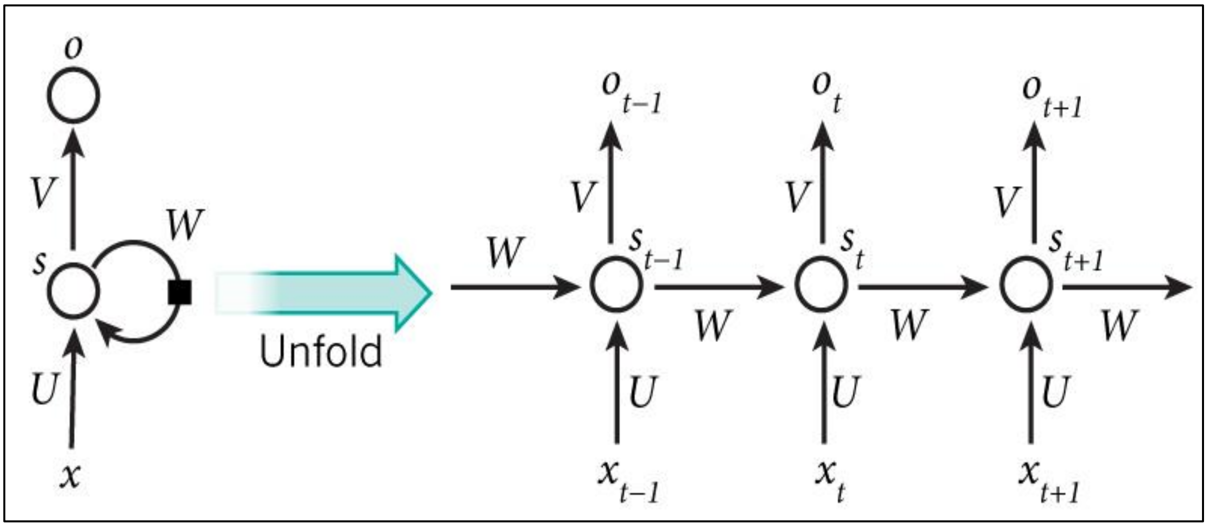


Figure 4. 1: RNN Network

The above model describes the content deployment of an RNN. Implementing here can be understood simply as drawing a sequential neural network. For example, if we have a sentence consisting of 5 words "Very handsome", the deployed neural network will consist of 5 layers of neurons corresponding to each letter. At that time the calculation inside the RNN is carried out as follows:

+ X t is the input at step t. For example, x_1 is a one-hot vector corresponding to the 2nd word of the sentence (*trai*).

+ S t is the hidden state at step t. It is the *memory* of the network s_t calculated based on both the hidden states ahead and the input at that step: s_t= f(Ux t + Ws_{t-1}). The function f is usually a non-linear function such as hyperbolic (fishy) tang or ReLu. To do the math for the first hidden element we need to initialize s-1, usually the initialization value is appended to 0.

+ O t is the output at step t. For example, if we want to predict the next word that might appear in a sentence, o t is a vector that determines the words in our vocabulary list: o_t= softmax(Vs_t).

- **About LSTM**

Every regression network takes the form of a sequence of repeated modules of a neural network. With standard RNN networks, these modules have a very simple structure, usually a fishy layer. LSTM also has such a chain architecture, but the modules in it have a different structure than the standard RNN network. Instead of having only one layer of neural networks, they have up to 4 layers that interact with each other in a very special way.

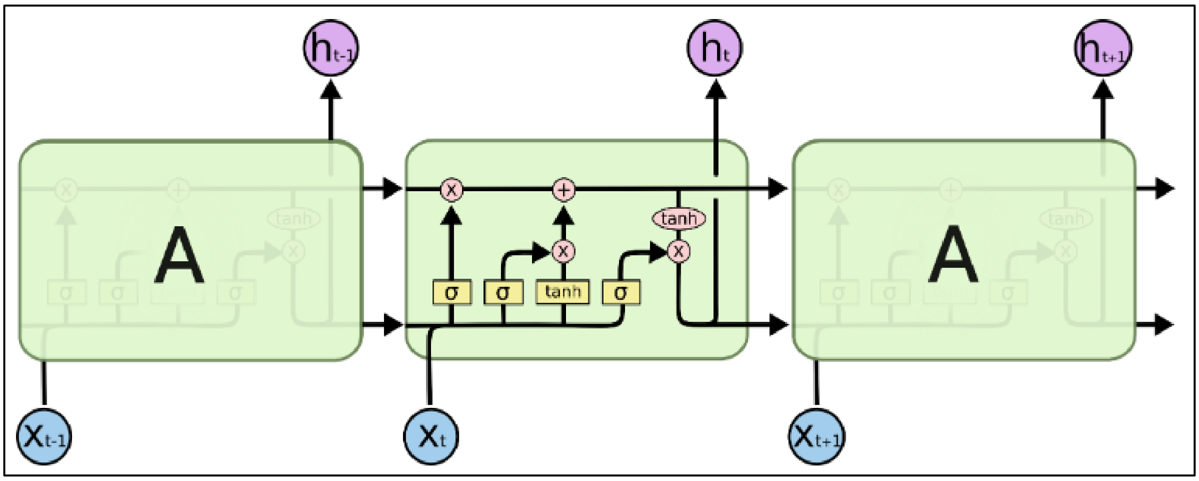


Figure 4. 2: LSTM structure

There is a repositioned LSTM database of a cell, an import port, an export port, and a forgetting port. The feature remembers the overtime, "time" time, and "time notifications" that determine the flow of information into and out of the database.
The LSTM system does not have enough LSTMs that have been implemented to work with the freeing and disappearing slope problems that have been encountered when working learning RNNs. Are defining modes to length, not the favorable time of LSTM on RNNs, unknown patterns and other character methods in other applications.

- History:

LSTM was proposed in 1997 by Sepp Hochreiter and Jürgen Schmidhuber. By introducing Constant Error Carousel (CEC) units, LSTM handles explosive and disappearing problems. The initial version of the LSTM block consisted of cells, input and output ports.

In 1999, Felix Gers and his advisors, Jürgen Schmidhuber and Fred Cummins introduced the forgetting gate (also known as the Gate Keeper) into the LSTM architecture, allowing LSTM to reset its own state.

In 2000, Gers & Schmidhuber & Cummins added peephole connections (cell-to-gate connections) to the architecture. In addition, the output trigger function has been omitted.

In 2014, Kyunghyun Cho et al. came up with a simplified variant called the Gated periodic unit (GRU).

Among other successes, LSTM achieved record results in natural language text compression, undifferentiated connected handwriting recognition, and won ICDAR's handwriting competition (2009). The LSTM network is a key component of the network achieving a record phonemic error rate of 17.7% on the classic natural speech dataset TIMIT (2013).

Since 2016, major tech companies including Google, Apple, and Microsoft have been using LSTM as a basic component in new products. For example, Google used LSTM for voice recognition on smartphones, for the Allo smart assistant, and for Google Translate. Apple used LSTM for the "Quicktype" function on the iPhone and for Siri. Amazon uses LSTM for Amazon Alexa.

In 2017, Facebook performed about 4.5 billion automated translations per day using short-term memory networks.

In 2017, researchers from Michigan State University, IBM Research, and Cornell University published a study during the Knowledge Discovery and Data Mining (KDD) workshop. Their study describes a new neural network, which works better on certain data sets than the widely used short-term memory neural network.

Furthermore, in 2017, Microsoft reported 95.1% recognition accuracy across the call center, incorporating a vocabulary of 165,000 words. The approach used "session-based long-term memory."

- LSTM's core ideas:

The key to LSTM is the cell state - the main horizontal line at the top of the diagram.

The cell state is a tape-like form. It runs through all the links (network nodes) and only interacts linearly slightly. Therefore, information can be easily transmitted smoothly without fear of being changed.

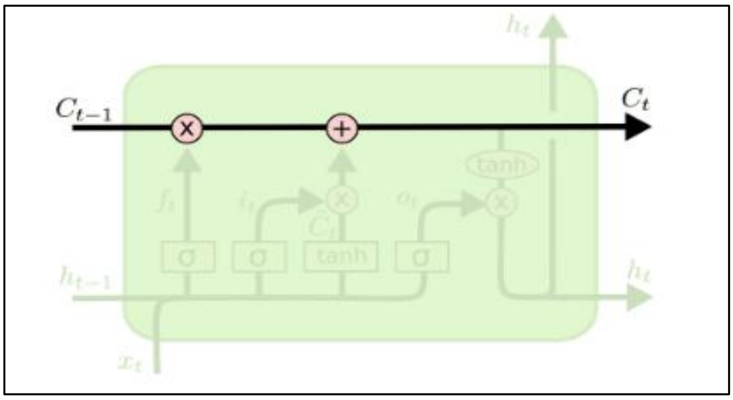


Figure 4. 3: Cell state

LSTM has the ability to remove or add information necessary for the state of affairs, which is carefully adjusted by groups called gates. Ports are where information screens pass through it, they are combined by a sigmoid network layer and a multiplication.

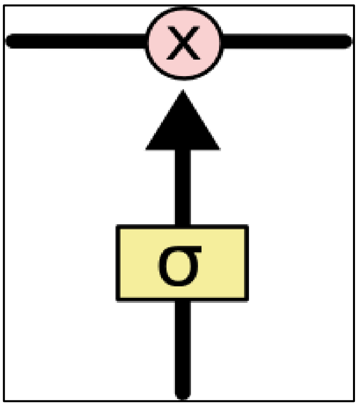


Figure 4. 4: Forgotten port

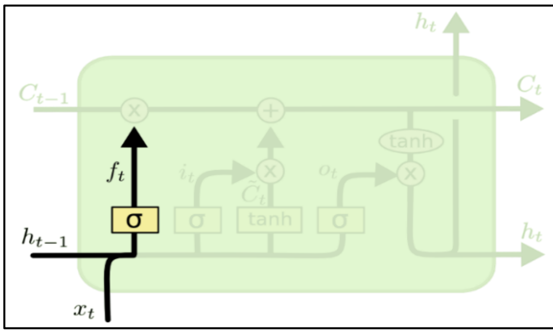
The sigmoid layer will give an output of a number in clause [0,1][0,1], describing how much information can be passed. When the output is 0 0, it means that no information passes through at all, and when it is 1 1, it means that all information passes through it.

An LSTM consists of 3 such gates to maintain and administer the state of the cell.

- Inside the LSTM:

The first step of LSTM is to decide what information to discard from the cell state. This decision is made by the sigmoid layer, called the "forget gate layer." It will take inputs of *ht*−1 and x_*txt* and result in a number in the range [0,1][0,1] for each number in the cell state *Ct*−1. The output is 1 1 indicating that it holds all the information, while 0 0indicates that all the information will be discarded.

Going back to the example of a language model that predicts the next word based on all previous words, with such problems, the cell state will probably carry information about the gender of a certain character that helps us use the correct personal pronouns. However, when referring to another person, we don't want to remember the gender of the character anymore, because it no longer works for this new subject.

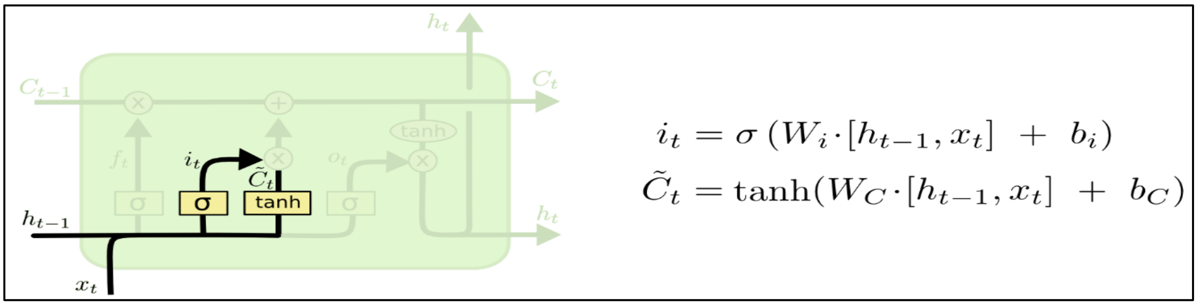


$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$

Figure 4. 5: Forgotten gate floor

The next step is to decide which new information we will save to the cell state. This consists of 2 parts. The first is to use a sigmoid layer called an "input gate layer" to decide which values we will update. This is followed by a fishy layer that creates a vector for the new value *Ct* to add to the state. In the next step, we will combine those 2 values to create an update to the state.

For example, with our language model example, we would want to add the gender of this new character to the cellular state and replace the gender of the previous character.



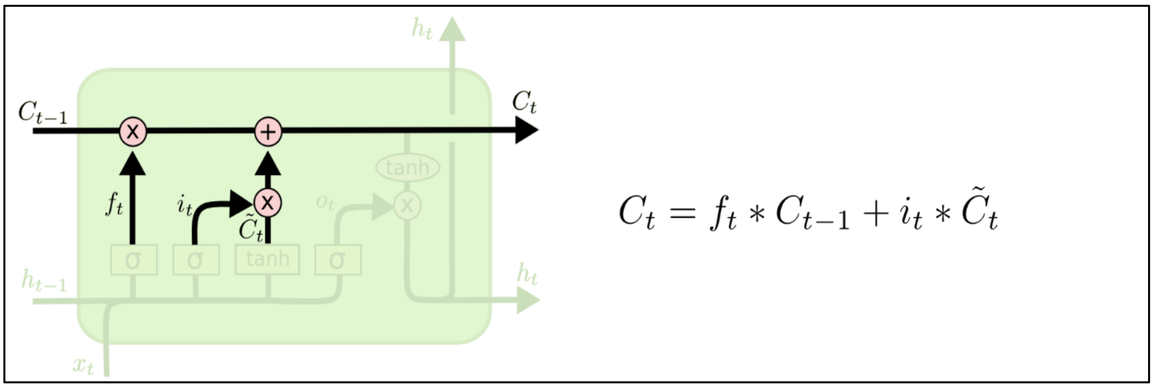
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

Figure 4. 6: Entrance floor

Now it's time to update the old cell state Ct−1 to the new state *Ct*. In the previous steps, we have decided what to do, so now we just need to do it.

We'll use the old state with *ft* to get rid of the information we decided to forget earlier. Then add *it*∗*Ct*. This newly acquired state depends on how we decide to update each state value.

With the language model article, it is about removing information about the gender of the old character, and adding information about the gender of the new character as we decided in the previous steps.

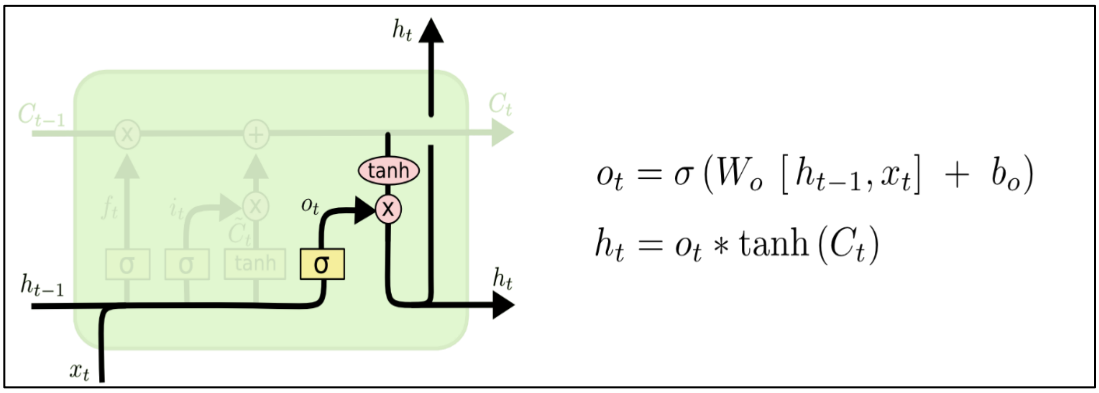


$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Figure 4. 7: Status updates

Ultimately, we need to decide what we want the output to be. The output value will be based on the cell state, but will be further screened. First, we run a sigmoid layer to decide which part of the cell state we want to output. We then take the cell state through a fishy function to get its value to about [-1,1][−1,1], and multiply it by the output of the sigmoid gate to get the desired output value.

With the example of a language model, just look at the subject where we can give information about an adverb that follows. For example, if the subject's output is singular or plural, we can tell what the form of the adverb that follows it should be.



$$o_t = \sigma\left(W_o \ [h_{t-1}, x_t] \ + \ b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Figure 4. 8: Determine the output

5. EXPERIMENTAL RESULTS

LSTM is a big step in the use of RNNs. Its idea makes it possible for all RNN steps to query information from a larger set of information. For example, if you use RNN to create a description for a photo, it can take a portion of the photo to predict the description from all the input words. Xu, et al. (2015) have done exactly this. There have also been many really interesting results to be noticed and there seem to be more results than we know.

- **Application**

Control the robot
Time series prediction
Speech recognition
Rhythmology
Musical composition
Learn grammar
Handwriting recognition
Recognition of human actions
Sign language translation
Protein homologous detection
Prediction of protein localization
Detection of time series anomalies
Some predictive tasks in the field of business process management
Predictions in the path of medical care
Semantic analysis
Audience segmentation

5. 1. Solution Model

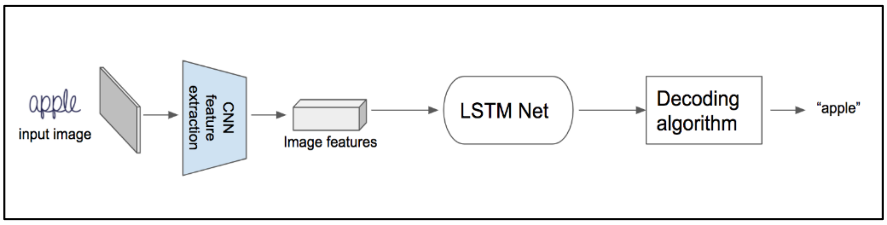


Figure 5.1: Solution model

First, the image is taken to CNN to extract the image features. The next step is to apply Recurrent Neural Network to these features followed by a special decoding algorithm. This decoding algorithm receives LSTM outputs from each time step and generates the final label.

The detailed architecture will be as follows: FC - fully connected layer, SM - softmax layer.

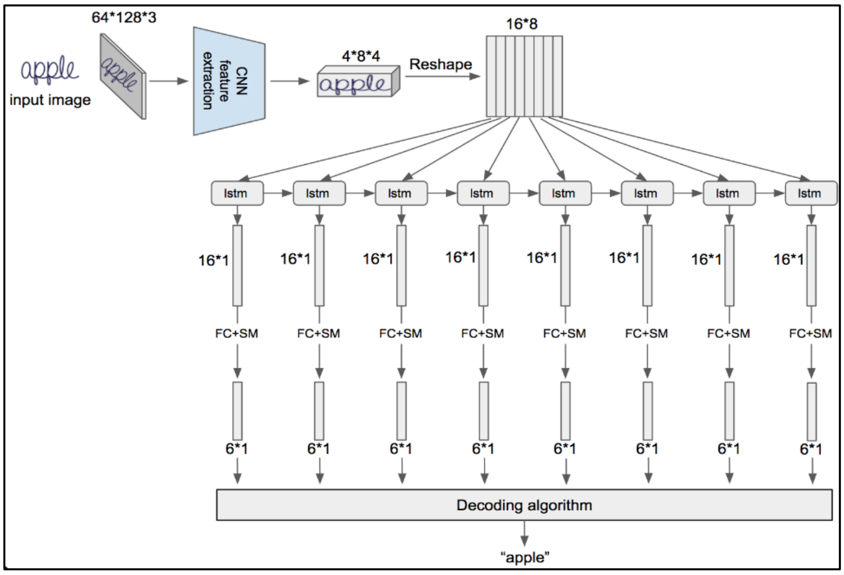


Figure 5. 2: Detailed architecture

The image has the following shape: height equals 64, width equals 128, and channel number equals 3.
The extracted input image featured CNN with a size of 4*8*4. Use the image of "apple" into the tensor application for easy understanding. The height is 4, the width is 8 (theseare spatial elements), and the number of channels is equal to 4. Therefore, it is necessary to convert the image from 3 channels to 4 channels.

Next is the step of performing a reshaping of the operation. The sequence of 8 vectors of 16 elements will then be obtained. Then these 8 vectors are fed to the LSTM algorithm and receive its output - which are also vectors of 16 elements. Then it is applied the fully connected layer followed by the softmax layer and takes the vector of 6 elements. This vector contains the observed probability distribution of alphabetical symbols at each LSTM step.

On the diagram above, there are 8 probability vectors at each LSTM time step. Let's trick the most probable symbol at each time step. The result is obtained a sequence of 8 characters - the most probable one letter at each step. Then, paste all consecutive repeating characters into one. In this example, two letters "e" are pasted into one letter. Special blank characters allow the separation of symbols repeated in the original labeling. Empty characters have been added to the alphabet to teach our neural networks to predict the space between such case symbols. Then remove all blank icons. Look at the illustration below:

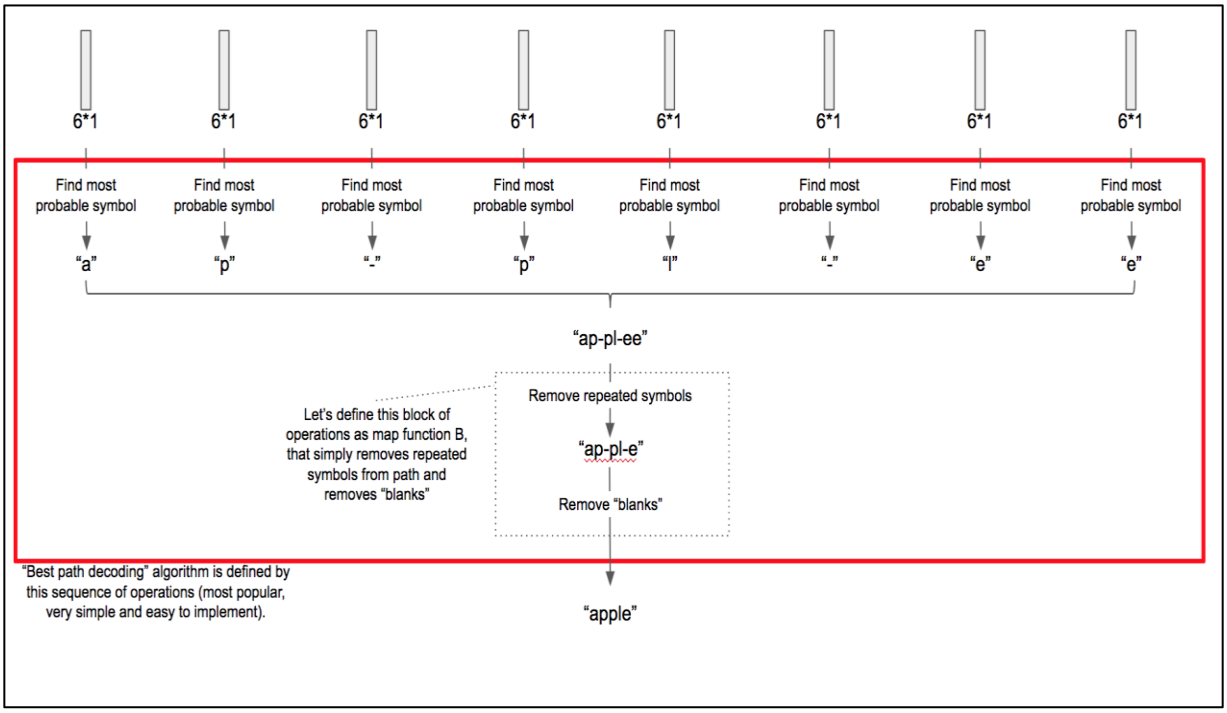

6*1
6*1
6*1
6*1
6*1
6*1
6*1
6*1
Find most probable symbol
"a"
"p"
"-"
"p"
"l"
"-"
"e"
"e"
"ap-pl-ee"
Remove repeated symbols
"ap-pl-e"
Remove "blanks"
Let's define this block of operations as map function B, that simply removes repeated symbols from path and removes "blanks"
"Best path decoding" algorithm is defined by this sequence of operations (most popular, very simple and easy to implement).
"apple"

Figure 5. 3: Separate each word

When it came to the network, it replaced the decryption algorithm with the CTC Loss layer.
A slightly complex Neural Network architecture is used in the implementation. The architecture is as follows, but the main principles remain the same.

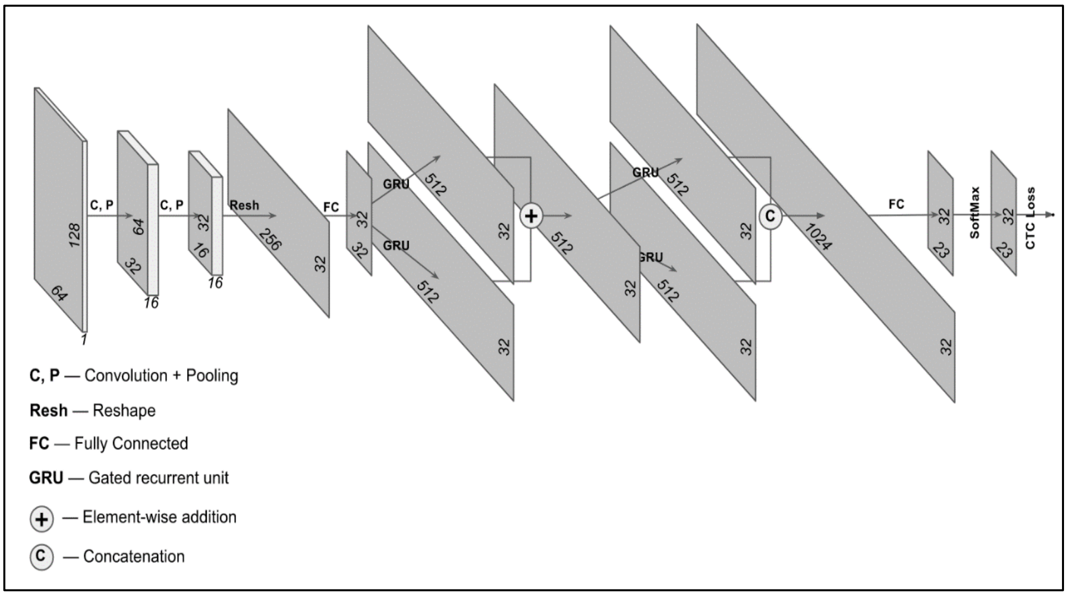
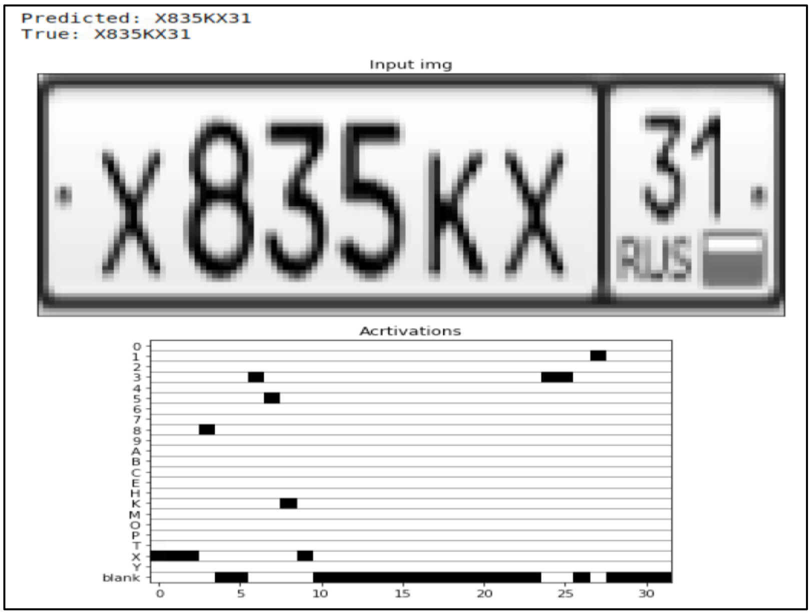


Figure 5. 4: Neural Network architecture

After training the model, it is applied on images from the test suite and has really high accuracy. It is visualized the probability distribution from each RNN step as a matrix. Here's an example.



```
Predicted: X835KX31
True: X835KX31
```

Figure 5. 5: Input image and probability distribution from each step

The rows of this matrix correspond to all alphabetic symbols plus "blank". The column corresponds to the RNN steps.

6. EXPERIMENT

6.1. Installation

Software requirements: Components: Ubuntu, GPU, Python, libraries such as keras, tensorflow, opencv, matplotlib,...

Test database: a dataset of 10,000 images identical to real number plates - from Supervisely where 95% of the dataset is for trains and 5% of the dataset is for testing.

Train consists of 10. 821 entries in each of the ANN and IMG folders.

The test consists of 561 entries in each ann and img directory.

The structure of the items will be as follows:

```
.├── data| ├── test
| | ├──ann
| | └──img
| └── train
| ├──ann
| └──img
└── img_ocr.py
```

Testing (the code will be detailed in the appendix):

+ Check the version of the keras and tensorflow libraries

+ Declare the necessary library

+ Get the alphabet: The machine will receive the alphabet in the training dataset. The license plate number in the training dataset consists of 8 letters and digits.

result:

```
Max plate length in "anpr_ocr__train": 8
Max plate length in "anpr_ocr__train": 8
Letters in train and val do match
Letters: 0 1 2 3 4 5 6 7 8 9 A B C E H K M O P T X Y
```

+ Input data generator:

result:

Text generator output (data which will be fed into the neutral network):

1) the_input (image)

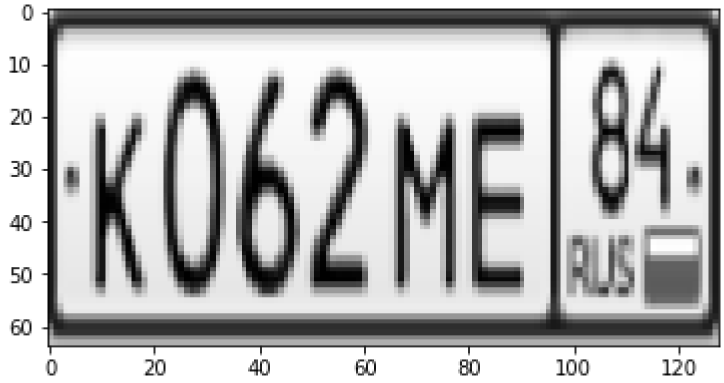


Figure 6.1: Input image

2) the_labels (plate number): K062ME84 is encoded as [15,0,6,2,16,13,8,4]

3) input_length (width of image that is fed to the loss function): 30 == 128 / 4 - 2

4) label_length (length of plate number): 8

+ Error calculation function and data set, neural network model

+ Describe and train models

+ This block will take about 30 minutes.

result:

```
____________________________________________________________________________________
Layer (type) Output Shape Param # Connected to
====================================================================================
the_input (InputLayer) (None, 128, 64, 1) 0
____________________________________________________________________________________
conv1 (Conv2D) (None, 128, 64, 16) 160 the_input[0][0]
____________________________________________________________________________________
```

```
max1 (MaxPooling2D) (None, 64, 32, 16) 0 conv1[0][0]
__________________________________________________________________________________________________
conv2 (Conv2D) (None, 64, 32, 16) 2320 max1[0][0]
__________________________________________________________________________________________________
max2 (MaxPooling2D) (None, 32, 16, 16) 0 conv2[0][0]
__________________________________________________________________________________________________
reshape (Reshape) (None, 32, 256) 0 max2[0][0]
__________________________________________________________________________________________________
dense1 (Dense) (None, 32, 32) 8224 reshape[0][0]
__________________________________________________________________________________________________
gru1 (GRU) (None, 32, 512) 83712001[0][0]
__________________________________________________________________________________________________
gru1_b (GRU) (None, 32, 512) 83712001[0][0]
__________________________________________________________________________________________________
add_1 (Add) (None, 32, 512) 0.1[0][0]
 gru1_b[0][0]
__________________________________________________________________________________________________
gru2 (GRU) (None, 32, 512) 1574400 add_1[0][0]
__________________________________________________________________________________________________
gru2_b (GRU) (None, 32, 512) 1574400 add_1[0][0]
__________________________________________________________________________________________________
concatenate_1 (Concatenate)(None, 32, 1024) 0–2
  gru2_b[0][0]
__________________________________________________________________________________________________
dense2 (Dense) (None, 32, 23) 23575 concatenate_1[0][0]
__________________________________________________________________________________________________
softmax (Activation) (None, 32, 23) 0.2
==================================================================================================
Total params: 4,857,319
Trainable params: 4,857,319
Non-trainable params: 0
__________________________________________________________________________________________________
Epoch 1/1
10298/10298 [==============================] - 1219s 118ms/step - loss: 0.3557 - val_loss: 1.7434e-04
```

+ Neural network output decoding function
+ Test test

Result:

Prediction: B331EC36
Fact: B331EC36

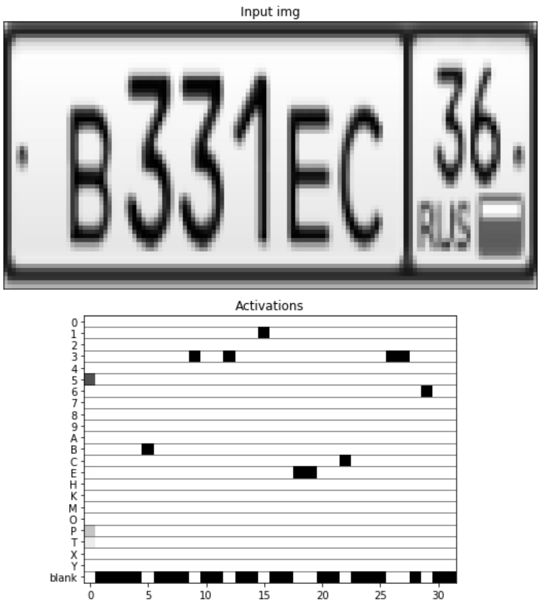


Figure 6.2: Image of input and probability distribution from each step

Prediction: K122CA31

True: K122CA31

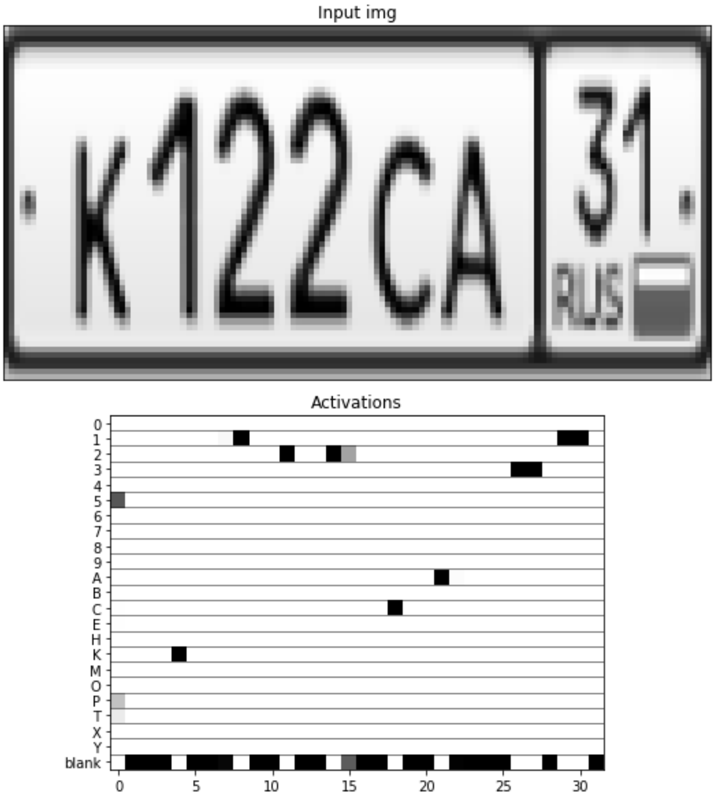


Figure 6.3: Image of input and probability distribution from each step

Prediction: M202XP51

Fact: M202XP51

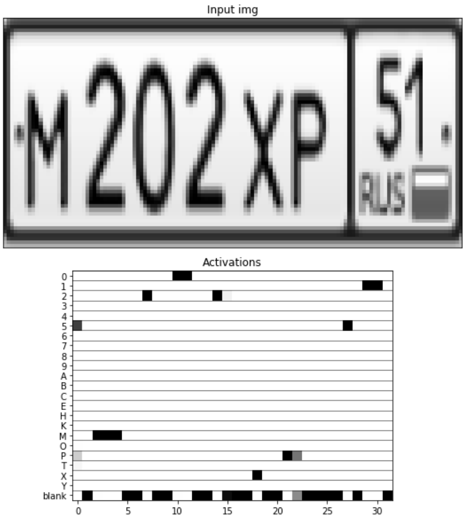


Figure 6.4: Image of input and probability distribution from each step

Prediction: T431KA04
Fact: T431KA04

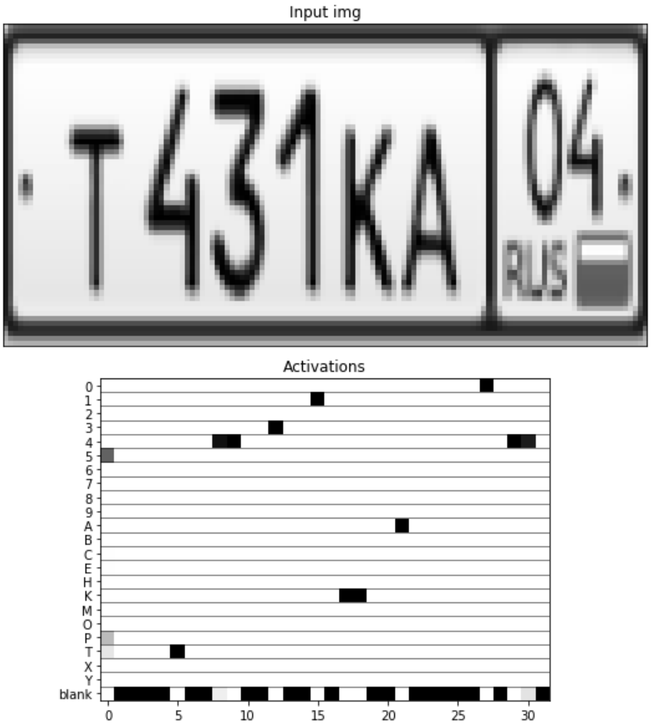


Figure 6.5: Image of input and probability distribution from each step

As a result, the sequence predicts with high accuracy.

6.2 Result of evaluation

- 95% train, 5% test result loss: 0.9749% - val_loss: 4.0664e-04
- 90% train, 10% test result loss: 1.0664%- val_loss: 5.1062e-04
- 85% train, 15% test result loss: 1.1132% - val_loss: 5.7124e-04

* Conclusion, comparison:

From the above results, it can be seen that with a data set of 95% train, 5% of tests give less loss recognition results as well as higher accuracy than other tests. Tests using deep learning techniques in optical character recognition showed better results (98%-99%) than the Support vector classfier (70% accuracy), Navier Bayes (40-70% accuracy), and C4.5 (60-70% accuracy) in some previous publications.

CONCLUDE

The problem of optical character recognition is always interested and developed by the ability to apply widely, especially in the digitization of data, storing documents dating back to ancient times, supporting the visually impaired or converting handwritten books into digital documents, After a period of research to solve the problem of optical character recognition, The thesis has achieved some results as follows:

- The thesis introduced the problem of optical character recognition, listing the applications of the problem with many different fields such as information storage, security, banking. At the same time, the thesis introduces the basic steps in the problem of optical character recognition. The main problems in optical character recognition when solving practical problems such as digitizing ancient texts.
- The thesis explores some popular optical character recognition solutions and software such as: Tesseract OCR, GOCR, FreeOCR, JavaOCR.
- At the same time, the thesis introduces basic knowledge about artificial intelligence, deep learning. The special thesis introduces in detail the Long short term memory (LSTM) deep learning method and the application of LSTM deep learning techniques in optical character recognition. The thesis was tested on a database of 10,000 photographs of license plates.
- Through testing, the thesis showed that the application of the LSTM method is quite effective in optical character recognition. However, experimentally there are some limitations such as it is more difficult to recognize smaller characters written on the same line.

ACKNOWLEDGMENTS

The authors would like to thank Van Lang University for supporting this research.

REFERENCES

1) Gupta, Maya R.; Jacobson, Nathaniel P.; Garcia, Eric K. (2007). "OCR binarisation and image pre-processing for searching historical documents" (PDF).
2) Suen, C.Y.; Plamondon, R.; Tappert, A.; Thomassen, A.; Ward, J.R.; Yamamoto, K. (May 29, 1987). Future Challenges in Handwriting and Computer Applications. 3rd International Symposium on Handwriting and Computer Applications, Montreal, May 29, 1987.
3) *Sarantos Kapidakis, Cezary Mazurek, Marcin Werla (2015).* Research and Advanced Technology for Digital Libraries. *Springer. p. 257.* ISBN 9783319245928.
4) "Code and Data to evaluate OCR accuracy, originally from UNLV/ISRI". *Google Code Archive.*
5) Riedl, C.; Zanibbi, R.; Hearst, M. A.; Zhu, S.; Menietti, M.; Crusan, J.; Metelsky, I.; Lakhani, K. (February 20, 2016). "Detecting figures and part labels in patents: competition-based development of image processing algorithms". International Journal on Document Analysis and Recognition.
6) Pati, P.B.; Ramakrishnan, A.G. (May 29, 1987). "Word Level Multi-script Identification". Pattern Recognition Letters. pp. 1218–1229.
7) Tappert, C. C.; Suen, C. Y.; Wakahara, T. (1990). "The state of the art in online handwriting recognition". IEEE Transactions on Pattern Analysis and Machine Intelligence.
8) Sezgin, Mehmet; Sankur, Bulent (2004). "Survey over image thresholding techniques and quantitative performance evaluation" (PDF). Journal of Electronic Imaging.
9) *Trier, Oeivind Due; Jain, Anil K. (1995).* "Goal-directed evaluation of binarisation methods" (PDF). *IEEE Transactions on Pattern Analysis and Machine Intelligence.*
10) Milyaev, Sergey; Barinova, Olga; Novikova, Tatiana; Kohli, Pushmeet; Lempitsky, Victor (2013). "Image binarisation for end-to-end text understanding in natural images" (PDF). Document Analysis and Recognition (ICDAR) 2013. 12th International Conference on.
11) A. F. Mollah, N. Majumder, S. Basu, and M. Nasipuri, "Design of an Optical Character Recognition System for Camera-based Handheld Devices," IJCSI, vol. 8, no. 4, pp. 283–289, 2011.
12) C. N. E. Anagnostopoulos, I. E. Anagnostopoulos, V. Loumos, and E. Kayafas, "A License Plate-Recognition Algorithm for Intelligent Transportation System Applications," IEEE, vol. 7, no. 3, pp. 377–392, 2006.
13) J. Diaz-escobar, "Optical Character Recognition based on phase features," IEEE, 2015.
14) M. Shen, "Improving OCR Performance with Background Image Elimination," 2015 12th Int. Conf. Fuzzy Syst. Knowl. Discov., pp. 1566–1570, 2015.

15) B. Jain and M. Borah, "A Comparison Paper on Skew Detection of Scanned Document Images Based on Horizontal and Vertical," IJSRP, vol. 4, no. 6, pp. 4–7, 2014.

16) Holley, Rose (April 2009). "How good can it get? Analysing and improving OCR accuracy in large scale historic newspaper digitisation programs". D-Lib Magazine. Retrieved January 5, 2014.

17) "Optical Character Recognition (OCR) – How it works". *Nicomsoft.com.* Retrieved June 16, 2013.

18) *Milyaev, Sergey; Barinova, Olga; Novikova, Tatiana; Kohli, Pushmeet; Lempitsky, Victor (2013).* "Image binarisation for end-to-end text understanding in natural images" (PDF). *Document Analysis and Recognition (ICDAR) 2013. 12th International Conference on.* Retrieved May 2, 2015.

19) "OCR Introduction".*Dataid.com.* Retrieved June 16, 2013.

20) Ray Smith (2007). "An Overview of the Tesseract OCR Engine" (PDF). Retrieved May 23, 2013.

21) "Train Your Tesseract". Train Your Tesseract. September 20, 2018. Retrieved September 20, 2018.

22) "Basic OCR in OpenCV | Damiles". Blog.damiles.com. November 20, 2008. Retrieved June 16, 2013.

23) Gupta, Maya R.; Jacobson, Nathaniel P.; Garcia, Eric K. (2007). "OCR binarisation and image pre-processing for searching historical documents" (PDF). Pattern Recognition. 40(2): 389. doi:10.1016/ j.patcog. 2006. 04.043. Retrieved May 2, 2015.

24) https://indatalabs.com/blog/ocr-automate-business-processes