Devising Muslim Name using Quantum Deep Learning

Syed Atif Ali Shah[1, 2, 3], Syed Asif Ali Shah [3], Usama Siddiqui [3]
[1] Department of CS, Bahria University, Islamabad, Pakistan.
[2] Department of CS, Al-Madinah International University, Malaysia.
[3] 3 Detechpro LLC , Integrating Solutions, Newark, Delaware, USA
welcomeatif@yahoo.com[1], syed.asif@detechpro.net[3], muhammad.siddiqi@detechpro.net[3]
Corresponding author: Syed Atif Ali Shah.

Abstract – This article proposes innovative quantum deep learning algorithms capable of generating Muslim names. The approach employs a Quantum Recurrent Neural Network technique as the deep learning model to produce new names by learning various name patterns. The researchers gathered thousands of Muslim names from well-known domains and curated the dataset, ultimately selecting 1000 pure and popular names for the training set after analyzing the full dataset. The provided system leverages several key features from classical quantum mechanical neural network libraries. Furthermore, the discussion indicates that applying this deep learning technique to larger datasets is expected to yield more accurate results, surpassing those of traditional machine learning algorithms. The validity of the proposed model is evidenced by the similarity between the artificial and real names.

Keywords—Quantum Recurrent Neural Network, Quantum Deep Learning, Quantum Neural Network, Quantum Artificial Intelligence, Muslim Person Name.

Introduction

The name of a person is the sweetest and most valuable sound of any language for this person. It is a method to identify a person. It eventually captures a combination of attributes when heard, especially when others know you and your personality. A name is important to everyone because it tells things about them (gender, religion, race, etc.). A person's given name (sometimes known as a first name) is a portion of their given name. Identifies a single individual and distinguishes him or her from other members of a group (typically, a family). For legal and administrative purposes, a person's entire name is frequently used to identify them. Some persons go by titles, nicknames, or other formal or informal designations and only use a portion of their complete name.

Individual qualities are influenced by the name of the person. Deep emotional, cultural, familial, and historical bonds hold them together. They also give us a sense of who we are, where we fit in the world, and what communities we are a part of. This is why blunders, mispronunciation of our preferred/common names, and gender misgendering can have a negative impact on our sense of belonging. Ethnic, familial, regional, and religious traditions influence naming practices in Pakistan. The majority of Muslim names have Arabic, Indo-Persian, or Turkish roots. The importance of tribal and family identities is highly valued. As a result, many people mix the names of their tribe with their families.

For example, due to Pakistan's large monotheistic religions population, traditional Arabic-Abrahamic names are the most frequent; (male) Ali, Muhammad, Yusuf, Bilal, and Hamza; (female) Noor, Mariam, Ayesha, and Fatima. As a last name, the child is usually given the father's most commonly used name. This is also true for some married women who adopt their husband's most common surname as their own. When a couple marries, Islamic law does not oblige the wife to take her husband's surname. As a result, a large number of Muslim women still use their maiden names The surname KHAN, which is common among Pashtun's, is one of the noticeable contrasts to this tendency. Another typical surname passed down through generations is SHAH. As a result of colonial influence, many Muslims have adopted multiple-style naming systems.

A name is unquestionably one of the most essential rights for human beings, as it should have an aesthetic quality and a socially accepted meaning, thus everyone should have a nice name. Personal names, such as a person's name, a family name, or even a nickname, are examples of human names. The name (Wahshaa) وحشة sends a message, whether positive or negative and has an impact on social interaction. One of the Islamic precepts is to provide a pleasant name to one's child. As a result, Muslim parents are forced to give their children a respectable name. "To whoever is born a boy and names him Mohammad solely for the love of one and the blessing name, then both he and his son," the Prophet Muhammad (PBUH) said, "To whoever is born a boy and names him Mohammad solely for the love of one and the blessing name, then both he and his son," because names influence a person's character (Raafat, 2004).

Personal names give their bearers confidence by communicating who they are to the entire world, as well as the occasion and origins of their names. From a semantic approach, names are studied. Although not all names have meanings, those that do have a positive connotation. The sociocultural backdrop, attitudes, beliefs, and physical surroundings are all nonlinguistic factors. Each civilization has its own set of rituals when it comes to naming its newborns. Parents appear to adhere to names that have been used within the same family, such as naming their children after grandparents, uncles, and other relatives, when it comes to naming their children. Finally, naming something is an act that reveals a variety of things, such as traditions, hopes, feelings, concerns, and everyday events [1]. Choosing a proper name for an infant is a thrilling experience; it is part of displaying your child's individuality, which they will carry with them until the end of their life. It can also be frightening, with so many child names to choose from and so many plans to consider. A name generator can help you in a variety of ways. It's simple to come up with name ideas by using a name generator. It only takes a few minutes to enter your keywords and get a list of possible names. Name generators use data from a large data collection to generate a list of probable names that are a good fit for the required features. This paper contributes by resolving an important issue of society, for which different techniques have been implemented, but none has shown appropriate results.

- Produced artificial names that sound similar to real names.
- Manual creation of dataset using real names.
- The data set does not contain all Muslim names. Some artistic names are real because they are not included in the data set. E.g Saleem, Aleem, Afiya etc.

This paper is organized into five sections. The first section describes the motivation and needs for such research in the introduction of the paper. The second section shows the existing work in the same field along with its pros and cons. The third section introduces tools and techniques for the implementation of the proposed model. The fourth section presents the results of the research. Finally, the fifth section, i.e. conclusion, concludes the process and need of this research.

Literature Review

The choice of the right name for a baby is an exciting experience. Research is always a time-consuming and difficult task. Various techniques are used to speed up the process. Several research activities are being carried out to provide a faster solution. Because [1] of the widespread usage of social networks and the widespread availability of the Internet connection, textual data on the Web has become significantly different from formal and standard data. In the processing of dialectal text from social media, deep neural network models have proven to be quite effective. In [2]; the authors add a Convolutional Neural Network to the existing LSTM neural tagging model for Arabic NER to extract significant character-level features. In many situations, our findings suggest that a character CNN can outperform previously employed character-level Bi-directional Long Short-Term Memory Networks. In another [3] study, the authors use character-based models to learn character patterns to predict the religion of individuals in South Asia, and researchers illustrate how to get around the supposed black-box nature of complicated nonlinear classifiers. In comparison [4] to English, Arabic is a morphologically rich language with little resources and a less explored syntax. For most evaluated Arabic NLP tasks, language-specific BERT models pre-trained on a large corpus achieve state-of-the-art performance. Recognition [5] of handwritten characters is a difficult study topic, and many studies have been published to recognize letters from many languages. This research provides a convolutional neural network for Arabic handwritten letter categorization, as well as optimization strategies for the best results. When employing word-level models in natural language processing, it's vital to keep in mind social media situations like Twitter, where words are spoken in a range of dialects and can be represented using character-level representations. To represent [6] and recognize Arabic text at the character level, a unique computer-aided deep learning Arabic text recognition system is proposed in this study. For document classification and sentiment analysis, the proposed system performs admirably.

Natural language processing [7] Processing faces additional hurdles as a result of the huge amount of textual data generated by social media. This research shows how deep learning models may be used to analyze these data and identify code-switching in dialectal Arabic in the dissertation. The authors [8] compare English and Arabic Named Entity Recognition in this study and find that the methodologies and characteristics employed in English NER work well in Arabic NER. The large [9] volume of unstructured data that is shared daily is driving up the demand for efficient information retrieval and extraction technologies. This paper offers a revolutionary deep learning technique for recognizing standard Arabic-named entities. Social media [10] media and internet access have resulted in a flood of information and textual data on the Internet in recent years, presenting new potential and difficulties for machine learning and natural language processing. In research [11]; the authors proposed adopting a deep learning strategy to handle the Arabic Named Entity Recognition issue, and utilizing an architecture that takes pre-trained word embedding and character-based representations as input, we got state-of-the-art results.

Quantum computers [12] are highly developed machines inspired by quantum physics [13]. Exploring the behavior [14]of atoms and particles is like inspecting and monitoring the behavior of quantum computers [15] by conducting quantum computers. This [16] is totally different from our ordinary computers or even [17] supercomputers. Saleem et al [18] presented a technique for EEG (Electo Encephalo Graphic) signals to be classified and analyzed using Quantum Inspired Wavelet Neural Network. Have got fine results as compared to existing techniques. Ali et al proposed [19] a quantum computing approach for personal authentication using finger knuckle print i.e FKP. FKP implemented Wavelet Neural Network, Quantum Neural Network, and Quantum Wavelet Neural Network. Research was supported by Hong Kong Polytechnic University. Saleem taha et al. [20] trained the Quantum Recurrent Neural Network (QRNN) with an auto-regressive model for the classification of EEG signals (Electro encephalon Graphy). As it was a hybrid model so they called it the Quantum Recurrent Neural Network Auto-Regressive (QRNN-AR) model. Research has shown a higher accuracy level along with a very short processing time. Ashwaq et al [21] presented a Quantum Neural Network on the early detection of disorders for the evolution of health care. Electro Cardio Graphy (ECG) signals are used as input data to train the QNN model. MR Taha et al [22] implemented a Quantum Radial Wavelet Neural Network model to classify and analyze Eeg signals. As compared to earlier models it was too complicated and slow, however, due to the Quantum Neural Network, has shown a faster processing time.

Tools and Technology

This section outlines the various methods and techniques that were utilized to implement the proposed model. It consists of three key phases: the Dataset and Pre-processing Phase, the Name Generation Process Phase, and the Building Blocks of the Model Phase. Each of these phases plays a critical role in the successful design, development, and deployment of the proposed model.

Dataset and Pre-processing Phase

This is the initial phase of the proposed model; in this phase there is a unit of different Muslim names, creating the lists of distinctive characters such as from (a-z), and calculating the words to make the name concise in size. There are (6474) characters, but (26) of them are unique. This procedure performs a role like to "end of sentence" symbol. But we are discussing only the Muslim names, not any end of sentences. We have a sequence of index characters from 0-26 by python dictionary also to map each character back to their corresponding position.

Name Generation Process

This section describes the second phase of the model in this phase recurrent neural network is deployed and trained on a given dataset. Different parameters are tuned up to improve the performance of the model. These parameters include the loss function, using forward propagation, gradients, backward propagation, explosion gradients, gradient clipping, and gradient descent for optimizations.

a. *Parameters Setup*

- Go through the improvement circle.
- Loss function calculation using Forward propagation.
- Calculating gradient according to loss function by Backward propagation
- Cut the gradients to avoid exploding gradients
- Using the gradients, revise our parameter with the gradient descent renew rule.
- Return the learned parameters.

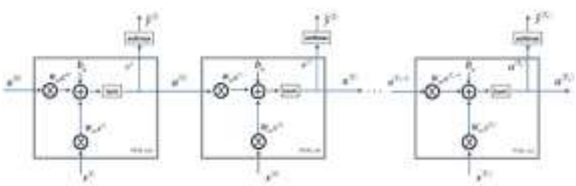

Figure 1: Operation of RNN Modules

Building blocks of the model

This is the third phase of the paper. it describes the architecture and process flow of the proposed model. Unlike other neural networks, RNN is not one go process. One module does not forget data once it is processed. Rather, it keeps the results and combines them with the next data.

Figure 1 illustrates the operation of various RNN modules. The RNN attempts to predict which character will be the next to give the preceding character at each step of the time. The dataset $\boldsymbol{X=(x^{(1)},x^{(2)},\ldots,x^{(Tx)})}$ is a list of characters in the training set, while $\boldsymbol{Y=(y^{(1)},y^{(2)},\ldots,y^{(Tx)})}$ is such that at every time-step tt, we have $\boldsymbol{y^{(t)}=x^{(t+1)}y^{(t)}=x^{(t+1)}}$.

We have constructed two major components of the overall model:

• Gradient clipping: used to prevent gradients from expanding.

• Sampling: a process for generating characters.

We then used these two functions to construct the model.

b. *Optimization loop (Gradient clipping)*

In this topic, we will look at how to create the clip function, which we called internal optimization in a loop. The entire loop structure is frequently repeated with a forward pass, a cost computation, a backward pass, and parameter updates. Indeed, when updating the parameters, we must implement gradient clipping to ensure that our gradient does not explode, which means using such large overloaded values.

We built a full-function clip below that makes dictionary gradients return a clip version of gradients if necessary. Different methods for recognizing gradients, a basic work methodology as needed, and an element-by-element clipping approach in which we offered the maximum value, i.e. 10, for every element that lies in between. If any component of a gradient vector is greater than 10, it is recognized as 10, but if any vector is less than -10, it is recognized as -10, and if it is between -10 and 10, it is recognized as left alone.

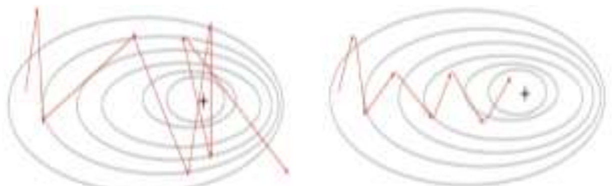

Figure 2: Visualization of a gradient, with and without gradient clipping.

In figure 2 gradient is described. The visualization shows the gradient decrease with and without the gradient clip, in the case of a modest "exploding gradient" problem in the network. To obtain the clipped gradients of our dictionary gradients, use the code below. Our function takes the highest threshold and returns the trimmed version. Here the gradients of "dWaa" = 10.0, "dWax" = -10.0,

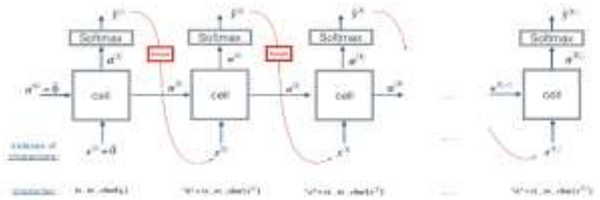

Figure 3: Trained Model of RNN

Figure 3 illustrates the trained model. To sample characters, use the sample function below. We must complete four steps:

- Step 1: Send the first "dummy" input to the network.
- Step 2: Perform one forward propagation step.

$$a^{(t+1)}=tanh(W_{ax}x^{(t)}+W_{aa}a^{(t)}+b) \quad (1)$$
$$z^{(t+1)}=W_{ya}a^{(t+1)}+b_y \quad (2)$$
$$\hat{y}^{(t+1)}=softmax(z^{(t+1)}) \quad (3)$$

- **Step 3**: Conduct sampling: Choose the next character's index based on the possibilities distribution.

  We've used the following strategy to determine the index that is consistent with the distribution:

  P(index=0)=0.1,P(index=1)=0.0,P(index =2)=0.7,P(index=3)=0.2.

"dWya"= 0.2971381536101662, "db" =10 and "dby" = 8.45833407 respectively.

c. *Sampling*

We assume that our model has been trained and that it will create new characters. The generation process is depicted in the diagram below.

• **Step 4:** The final step to implement in a sample() is to overwrite the variable x, which now stores x(t) at the cost of x(t+1). We created x(t+1) by constructing a one-warm vector identical to the individual we chose as our prediction. We must then advance propagate x(t+1) in Step 1 and continue repeating the technique until we obtain an "n" man or woman, signifying that we have reached the end of the Muslim name, Developing the language model

It is now time to create a character-level language model for text production.

Gradient descent

In this section, we implement a single-step stochastic descent gradient, where we run through certain training examples once at a time, therefore the optimization algorithm may be 50,50 approximately descending gradient.

We certainly have some steps of a common optimization loop for an RNN:

d. *Use forward propagation across the RNN to compute the loss.*

e. *The backward propagation through time was carried out to compute the gradients of the loss with respect to the parameters.*

• Clip the gradients if necessary

• Clip the gradients if necessary

• Update our parameters using gradient descent

f. *Implement this optimization procedure.*

Performs the following functions:

def rnn_forward(X, Y, a_prev, parameters):

def rnn_backward(X, Y, parameters, cache):

def update_parameters(parameters, gradients, learning_rate):

Quantum Computing

Quantum computers are advanced machines inspired by quantum physics. The study of the behavior of atoms and particles; so that quantum computers operate by studying and controlling the behavior of these particles. Which in a way completely is different from our conventional computers or even supercomputers. It is an upgraded and powerful version of these computers, but not exactly the next-generation computers. The computer took bits of zero and one and depends on only these two results. The quantum computer, on the other hand, will also have these possibilities, including the possibility of a mixture of both zero and one. For example, with some portion of zero in some portion of one and do to the outcome will still be there between both two possibilities. This property of superposition and entanglement of quantum physics empowers quantum computers to handle operations at speeds that are orders of magnitude faster than traditional computers while using a fraction of the energy. Quantum computers aim to revolutionize future quantum technologies and just like other transformations around the world. Quantum computers have the potential to impact our lives in so many ways like our security needs health care needs etc. It will be easier for quantum computers to study each property of atoms and molecules, as it works on a similar concept. That makes quantum computers different from supercomputers in that atoms and molecules be uncertain of outcomes. Due to its current benefits, it could be connected to more than 17 billion devices. Local quantum computers are the future of our coming generations.

Quantum Recurrent Neural Network

From the above discussion, we have concluded the working of RNN for producing names. Here, we are adopting quantum computing technology to resolve the same problem. QRNN are cousins of RNN, performing the same task in quantum computing. Quantum neural networks are tools for decision-making and inference. The data can only be encoded into qubits. The model of quantum neural networks is based on the multi-level activation function using the concept of quantum superposition state in quantum theory. Quantum cells and neurons can express more states than traditional signal functions, each with different quantum intervals. The quantum interval has been updated to map different data on different step-ladders. Fig. 4 shows the architecture of the QRNN, where X_1, X_2 through X_{ni} are inputs to the QRNN.

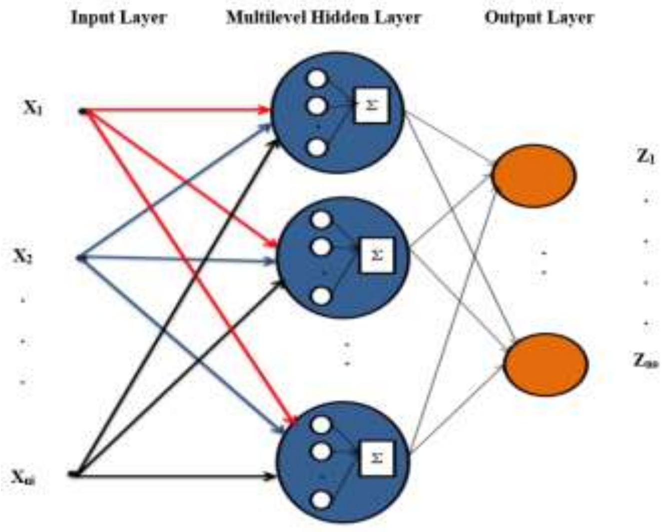

Figure 4: Trained Model of QRNN

Here the basic unit is qubit instead of bit; that is why our focus is to process qubit at each state. To comprehend the model, different inputs are represented by different colors. Then we have a multilevel hidden layer, represented with blue-color circles. These circles show the core functionality of the QRNN. This work builds the first quantum recurrent neuronal network (QRNN), which can demonstrate non-trivial tasks such as sequence learning and integer numerical classification. QRNN cells are constructed of parametric quantum neurons and combined with amplitude enhancement, create nonlinear activations of input polynomials and hidden states. We can extract the probability distribution on the predicted class at each step which allows us to extract the probability distribution at each step. The operation is the exact copy of the process performed in Fig. 4, however, the organization of the process is entirely different. One is traditional computing and another is quantum computing, that is why the way of processing is utterly different. Finally, orange circles show the output, i.e., Z_1 to Z_{no}.

To investigate the performance of this model, we provide Python implementations that enable the relatively efficient optimization of tens of thousands of parameterized quantum circuits. This shows that the model does not seem to be affected by the disappearing gradient problem, which plagues many extinguished quantum classifiers and classical RNNs. We define QRNN training systems by benchmark optimization hyper parameters and analyze network topologies suitable for simple memory and sequence prediction tasks. As described in Elman's seminal paper (1990).

Results

This section explains the results of the experiments conducted using the techniques mentioned earlier. Both methods have produced promising results. Due to the unavailability of quantum machines, we are unable to fully explore the potential of quantum computing. However, we can still practice and test many quantum algorithms using GPUs.

The results presented here are based on the Recurrent Neural Network model, as both models have yielded similar outputs. We used a single name from the comprehensive set of Muslim names as a training example and fed numerous random names into the algorithm to observe its behavior. After the model has learned the name transformation, stochastic gradient descent visits each example in a random order. Initially, the model may output random-looking and nonsensical characters, but it learns to generate good names after several thousand iterations. Table 1 summarizes the different iterations and their corresponding outputs, providing a detailed view of the model's performance at each stage. Additionally, a loss value is calculated at each iteration, allowing us to track the model's convergence and assess its overall effectiveness.

Table 1: Iterations with outputs

ITERATION: 0, LOSS: 65.867300
NKZXWTDMFQOEYHSQWASJJJVU, KZXW TDMFQOEYHSQWASJJJVU, DMFQOEYHS QWASJJJVU, EB, ZXWTDMFQOEYHSQWAS JJJVU, XWTDMFQOEYHSQWASJJJVU, B, MFQOEYHSQWASJJJVU, WTDMFQOEYHS QWASJJJVU, TDMFQOEYHSQWASJJJVU, FQOEYHSQWASJJJVU, KNEB, QOEYHSQW ASJJJVU, DUVGBAMWKWGFLYNFCZCHRVBBXVN ENGDQOXMIFPQMEPFVRMKBDP, NEB, UVGBAMWKWGFLYNFCZCHRVBBXVNE NGDQOXMIFPQMEPFVRMKBDPP, VGBAMWKWGFLYNFCZCHRVBBXVNEN GDQOXMIFPQMEPFVRMKBDPPW, GBAMWKWGFLYNFCZCHRVBBXVNENG DQOXMIFPQMEPFVRMKBDPPWE, BAMWKWGFLYNFCZCHRVBBXVNENGD QOXMIFPQMEPFVRMKBDPPWEL
ITERATION: 2000, LOSS: 24.567757
ILYSSOAL, DFA, EYSSOAL, ID, YSSOAL, A, TSNAL, SNAL, PAL, AROA, AL, QSA, HANIBWARIS, SA, ANHAYAQIS, A, KHAYA QIS
ITERATION: 6000, LOSS: 15.722399
KHUSON, HBDA, HUSONAD, LABADLR, W USIDFAKJAS, ABADNOH, SSKEEENA, AADMQA, SIDIFAD, ADIR, SALEEN, ARNA, ALEEM, SOEEBDULSEEKUM, JANIEYAHA SANEELIR, SHABENAN, AMIEYAINLANEE LMSALNEYA, BABGRAR, OMBULMHR, ABELEQA
ITERATION: 8000, LOSS: 14.992644
MHUSMIBAFMHAYAN, HEEB, HUSQIBAD MAHR, MAD, WUSLA, AA, TUSAL, AADISA, SOEL, ABIR, SAMAL, ARNA, ALEFA, SUL, K ANIDSHONN, SAD, ANIEY, BAAMILAH, RIFULAIS, AAMILAGDOVLAAZ
ITERATION: 10000, LOSS: 14.553417
KHUSQID, HAGA, HUSOM, KABA, WUSID, ABAAQIED, SNIDA, AADIR, SHAID, ABIR, SALEEN, AQOED, AKEEM, SKABAHIMFAH GSHAAZ, IHIM, SABAHIMAADATABAZA, AMIDN, ABAIR, NIDULINKANIF, AALMAN
ITERATION: 12000, LOSS: 14.208120
KAZMOR, EEEB, FUSMID, KABABISA, WO USAJAHAAYAM, ABABLULAAMILA, SMIDA , AADIR, SHAJALEEWAMAT, ABEEL, SALE EM, ARLA, ALEEM, SKABALMASABDULHAZAEMA, IJHEEYA, S ABAISAR, AMHAY, ABALRASABDULIAZADAD, NGAZANA, AA KMAM

ITERATION: 14000, LOSS: 13.977406
NESNIN, JHAB, KUSOR, NAD, WTNNA, AB ABNOEEBATASHA, SORAH, AADMA, SIDH A, ABISA, SALEEN, ARRA, AMEER, SOEEA HA, MANIAT, SADAIR, ANIDULNASAR, DA BIS, SHAZA, ABDULSAFIYA
ITERATION:56000, LOSS: 12.921273
INWEQ, AID, AZSHA, ILA, WUSNA, AABAN IL, SOUHAAM, AADIRA, SHAHAN, ABDULB AIYAR, SAIDA, ATMAA, ADDOR, SMEEA, H AMI, SADANA, AMILA, ABDURQUHADR, NOOZINAH, AAISHI
ITERATION:58000, LOSS: 12.855027
ABRURSALEEM, ABDUDKHMEE, AZMAR, ABDA, WISHA, AABELQA, SHEEM, AABIR, SHAHAL, ABDULA, MALEHA, ALMA, ABDU LIZEEMA, MUHA, ABDULWAKIS, SADAJR, ABDUSMALIAL, ABDUSMUH, IMBR, AAFR
ITERATION:60000, LOSS: 12.719778
MSUZTFANAIR, KALD, KUSRIF, MAD, YUS MAN, BABIRAF, WISAM, AAFIYA, UNAN, ABFIR, SALIM, AWRAABAR, ALEEM, SQEE ANA, MARIATATKHAQAH, SADANIMAH, BISAZAIR, HADIR, SHAYAN, AALILH
ITERATION:62000, LOSS: 12.720910
IMUSSDAAIM, AILA, BUZYAL, IMA, WASIL A, AABARIN, SHEER, AADIYA, SADIAM, ABELA, SAMEEN, AMRAA, AFEEM, SHAD, HAMIAULLEN, SADAMIBIMAHAH, ALFAWANNI, ABBAR, MAHTAR, AAMMAN
ITERATION:64000, LOSS: 12.627332
MAZNI, ISHA, KSSIR, MADAAR, WISHAD, A BABIR, TOSHID, AABIR, SHAHAM, ABERA, SALEEM, ANTAA, ADEEM, SIFA, MASHEN, SABAL, AIN, BADISAH, RAFYA, ABDULRAAHID

Here, the process started with iteration no. 0 having a loss of 65.86, and in iteration no. 1, the loss reduced to 24.56. Thus, as the number of iterations increases, the loss decreases. However, it is also observed that at iteration no. 56,000, the loss becomes 12.92, and after that, no significant reduction is observed. For example, in iteration no. 62,000, the loss remains the same at 12.72, with almost negligible change. Therefore, further iterations beyond this point would be a waste of time.

Conclusion

This paper proposed a model that generated a variety of Muslim names. Initially, it produced random symbols, but finally provided us with an attractive set of Muslim names. By running the algorithm, we can obtain even extremely good results. Here we have some really cool names like Shahan, Saleem, Aleem, Salim, Rafya, Sameen, Saida, Wasila, Yusman, Wisam, Aafiya, Unan, Adeem, Sifa, Mashen, Sabal, and Aalilh. The system has also learned that Muslim names may tend to start with Abdulizeema, Abrursaleem, Abdudkhmee, and so on. While some uncool names are being produced, this does not imply that our method isn't working; after all, not all Muslim names sound equally appealing. Aafa is a real Muslim name that is included in the training set. This approach should provide us with a list of the trendiest names from which to choose. Due to the relatively modest dataset used in this investigation, we were able to chain an RNN quickly on a CPU. In contrast, training an English vocabulary model would necessitate a significantly larger dataset, much more computation, and several hours of processing time on GPUs.

References

[1] L. Lulu and A. Elnagar, “Automatic Arabic Dialect Classification Using Deep Learning Models,” *Procedia Comput. Sci.*, vol. 142, pp. 262–269, Jan. 2018, doi: 10.1016/J.PROCS.2018.10.489.

[2] R. Chaturvedi and S. Chaturvedi, “It’s All in the Name: A Character Based Approach to Infer Religion,” *SSRN Electron. J.*, 2022, doi: 10.2139/ssrn.4012431.

[3] “Character convolutions for Arabic Named Entity Recognition with Long Short-Term Memory Networks - ScienceDirect.” https://www.sciencedirect.com/science/article/abs/pii/S0885230818301657?via%3Dihub (accessed Apr. 24, 2022).

[4] A. Y. Muaad, H. Jayappa, M. A. Al-Antari, and S. Lee, “ArCAR: A Novel Deep Learning Computer-Aided Recognition for Character-Level Arabic Text Representation and Recognition,” *Nov. Deep Learn. Comput.*, 2021, doi: 10.3390/a14070216.

[5] N. Wagaa and H. Kallel, “Convolutional neural networks,” *Appl. Data Anal. Princ. Appl.*, vol. 2022, no. iv, pp. 185–203, 2020, doi: 10.4018/978-1-

6684-2408-7.ch077.
[6] Shah, Syed Atif Ali, et al. "Generating Muslim Name using Character-Level Language Model in Deep Learning." *2023 International Conference on Smart Computing and Application (ICSCA)*. IEEE, 2023.
[7] M. Gridach, “Character-Aware Neural Networks for Arabic Named Entity Recognition for Social Media,” pp. 23–32, 2016.
[8] W. Antoun, F. Baly, and H. Hajj, “AraBERT: Transformer-based Model for Arabic Language Understanding,” 2020, [Online]. Available: http://arxiv.org/abs/2003.00104.
[9] A. I. Alharbi and M. Lee, *Combining character and word embeddings for affect in arabic informal social media microblogs*, vol. 12089 LNCS. Springer International Publishing, 2020.
[10] C. Sabty, I. Mesabah, Ö. Çetinoğlu, and S. Abdennadher, “Language Identification of Intra-Word Code-Switching for Arabic–English,” *Array*, vol. 12, no. November, p. 100104, 2021, doi: 10.1016/j.array.2021.100104.
[11] M. M.Abdelgwad, T. H. A Soliman, A. I.Taloba, and M. F. Farghaly, “Arabic aspect based sentiment analysis using bidirectional GRU based models,” *J. King Saud Univ. - Comput. Inf. Sci.*, 2021, doi: 10.1016/j.jksuci.2021.08.030.
[12] W. Hu, “Feasible Architecture for Quantum Fully Convolutional Networks,” 2021.
[13] S. Oh, J. Choi, and J. Kim, “A Tutorial on Quantum Convolutional Neural Networks (QCNN),” *Int. Conf. ICT Converg.*, vol. 2020-Octob, pp. 236–239, 2020, doi: 10.1109/ICTC49870.2020.9289439.
[14] D. B. Yousef, “Quantum-inspired NN for IR.”
[15] T. Hur, L. Kim, and D. K. Park, “Quantum convolutional neural network for classical data classification,” *Quantum Mach. Intell.*, vol. 4, no. 1, 2022, doi: 10.1007/s42484-021-00061-x.
[16] C. Zhao and X. Gao, “QDNN : deep neural networks with quantum layers,” vol. 2, 2021.
[17] A. Chalumuri, R. Kune, and B. S. Manoj, “ScienceDirect Training Training an an Artificial Artificial Neural Neural Network Network Using Using Qubits Qubits as Artificial Neurons : Quantum Computing Approach as Artificial Neurons : A Quantum Computing Approach,” *Procedia Comput. Sci.*, vol. 171, no. 2019, pp. 568–575, 2020, doi: 10.1016/j.procs.2020.04.061.
[18] S. M.R.Taha and Z. K. Taha, “Analysis and Classification of EEG Signals based on a New Quantum Inspired Wavelet Neural Network Model,” *Int. J. Comput. Appl.*, vol. 92, no. 5, pp. 23–30, 2014, doi: 10.5120/16005-5008.
[19] A. S. Altaher and S. M. R. Taha, “Personal authentication based on finger knuckle print using quantum computing,” *Int. J. Biom.*, vol. 9, no. 2, pp. 129–142, 2017, doi: 10.1504/IJBM.2017.085676.
[20] S. M. R. Taha and Z. K. Taha, “EEG signals classification based on autoregressive and inherently quantum recurrent neural network,” *Int. J. Comput. Appl. Technol.*, vol. 58, no. 4, p. 340, 2018, doi: 10.1504/ijcat.2018.10017211.
[21] A. N. Hassan, M. J. Mohsin, and A. H. K. Khwayyir, “Discrimination of ECG signal based on S-interpolation and quantum neural network,” *Ann. Trop. Med. Public Heal.*, vol. 23, no. 10, 2020, doi: 10.36295/ASRO.2020.231039.
[22] S. M.R.Taha and A. K. Nawar, “A New Quantum Radial Wavelet Neural Network Model Applied to Analysis and Classification of EEG Signals,” *Int. J. Comput. Appl.*, vol. 85, no. 7, pp. 7–11, 2014, doi: 10.5120/14851-3216.