Next Word Prediction Model Using Machine Learning

Abhishek Bhau
school of computing science and engineering
Galgotias University, greater noida, India
abhishek.22scse010220@galgotiasuniversity.edu.in

Danish Manawar
school of computing science and engineering
Galgotias University, greater noida, India
danish.22scse1010220@galgotiasuniversity.edu.in

Anjali Kapoor(Guide)
school of computing science and engineering
Galgotias University, greater noida, India
anjali.kapoor@galgotiasuniversity.edu.in

***Abstract*—Next-word prediction models are central to many natural language processing (NLP) tasks, including text completion, machine translation, and conversational AI. These models leverage a variety of machine learning techniques to predict the most likely next word based on the context of preceding text. This paper explores the evolution of next-word prediction models, from traditional n-gram models to modern transformer-based architectures, such as OpenAI's GPT and Google's BERT. We discuss the underlying methodologies, advancements in model architectures, and evaluation metrics used to assess their performance. Additionally, we review real-world applications and future directions for research in this domain.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

Next-word prediction has been a crucial component of natural language processing for decades, enabling advancements in many NLP applications such as auto-completion, speech recognition, and dialogue systems. Given an input sequence of words, the task of next-word prediction is to predict the most probable word that will follow the input sequence. Early approaches were based on statistical models such as n-grams, but more recent methods utilize deep learning models that are capable of capturing much richer contextual relationships.

This paper aims to review the key methodologies, architectures, challenges, and applications associated with next-word prediction.

A. Background

Traditional Methods: N-gram Models In early approaches, n-gram models were widely used for next-word prediction. An n-gram model is based on the assumption that the probability of a word depends only on the previous n-1 n1 words, i.e., it calculates the conditional probability P(wtwt1,...,wtn+1)P(wt wt1,...,w tn+1.

Identify applicable funding agency here. If none, delete this.

II. LITERATURE REVIEW

This section explains how to conduct a standard literature review. This paper is for seeking to break free from the schooling challenge distribution.The multi-window convolution algorithm (MRNN) is released, along with a connected remnant minimum gated unit (MGU), which is a condensed version of the LSTM on this cnn tries to skip some layers while training, resulting in significantly less training time and higher accuracy. The use of multiple layers of neural networks can purpose prediction delays n range names.RNN allows code consumers to rely on the following syntax to finish the code hassle. In comparison to existing methodologies, the authors claim that their methodology could be highly correct.Inner-vocabulary names and identifier prediction are examples of word prediction in elements. To predict terms within words, they used the LSTM neural language model. At the objective forecast, the community network model is proposed. The Bangla language was worked on by the authors. They recommend a unique method of anticipating the loss of words and terms. They advise using an N-gram-based whole language model, which predicts a set of phrases. They have reaped remarkable benefits. The authors utilised LSTM to predict the following word in Assamese. Maintain note language in accordance with the International Phonetic Association (IPA) chart and provide an example.

Humans with physical limitations in Baka form. This model employs Unigram, Bigram, and Trigram approaches to forecast the following phrase, and the prediction score is converted into the received phrase, however the accuracy drops to 30-40 .Because of the absence of gradients and complexity, it was discovered that obtaining acceptable accuracy via RNN method is difficult due to the disappearance of gradients and complexity Continuous NN takes more time to teach and evaluate in this paper.They used to be anticipating the subsequent letter of the Indian alphabet (PNCH), which became more to correct the textual content and a little bit regarding forecasting the subsequent phrase, however it turned out to

be absolutely correct to recognise.A method known as hit and miss is used, but the accuracy is low, and the model has stopped operating on this type of problem statement. This has been the most popular method of dealing with this type of problem; the paper explains LM and the confusion algorithm, which is the foundation of natural language processing; this allows us to generate 3-D input data for our model.The 1-degree sample characteristic function set of rules is utilised for extinction problem selection, but with as little accuracy as possible because the fundamental sentence became employed in teaching and comparing comparable statistics .

III. METHODOLOGY

The steps in methodology are:- .Data pre-processing .Comunity of LSTM

.DATA PRE PROCESSING

This painting depicts a versatile version that can also help users construct an indisputable model that can assist customers in detecting the next word. When creating the user.We would offer letters to the records set first and foremost, and then the data set would provide letters through which we should expect certain phrases. This sentence might be surpassed by the LSTM neural community and later it would expect the same the rating would then be transmitted via the same LSTM and later it would forecast the next word, as demonstrated in the above parent supplying input to the statistics. The neural network structure and our use of the Tensorflow library are shown below.The output layer with the same node as the input layer. First, introduce our helpful model. After several modules have been imported, Numpy pandas will import Nietzsche's default textual material. It is our information system.Step 2: The LSTM path is cell explicit, including a line that passes through the optimal point of frame.The transit line iscomparable to the cell kingdom.With only a slight direct touch, it turns instantly throughout the chain.It is straightforward for facts to unfold when they are unaltered.LSTM can delete or upload data to the telephone country, which is strictly monitored by entryways.Information can be bypassed via gates.They are the result of sigmoid neural network layers and a point repetition assignment. The sigmoid layer displays numbers somewhere within a clean and single list, indicating that the overall value must pass. The values of 0 way "do not permi anything," while the values of 1 way "permit it's all over!" For safety and to manipulate cell popularity, LSTM features those gates.

Tokenization is the process of breaking down huge text facts, essays, or corpora into smaller chunks. Smaller documents or traces of text can be used to create these smaller chunks.They can also serve as a word dictionary.The Keras Tokenizer allows us to vectorize a textual content corpus by turning each textual content element into a token.content into either a set of integers (each integer being the index of a token in a dictionary) or a list of stringsinto a vector with a binary coefficient for each token, based solely on phrase count,tf-idf is used. To learn more about the Tokenizer's magnificence and content, click here.Go here for information pre-processing using Keras.the texts will subsequently be converted into sequences.This is a method of converting textual content data to numbers so that we may perform more advanced studies on it. The education dataset will then be created. The 'X' will be made up of education statistics plus textual input..The outputs for the education records will be included in the 'y'. As a result, the 'y' includes all of the next phrase predictions for each 'X'.

The vocab size will be calculated using the period taken from the tokenizer.After that, submit 1 to Word index. We're including 1 since zero is allocated for padding, and we want to start counting from one.This will come in handy with our categorical crossentropy loss. The following is the rest of the code for tokenizing records, expanding the dataset, and converting the prediction set into express facts:It should be noted that pre-processing can be improved. You can experiment with various approaches to improve the pre-processing step, which may aid in achieving higher loss and accuracy in fewer epochs.

IV. NEUTRAL NETWORK MODELS

To overcome the limitations of n-gram models, researchers began exploring neural networks for language modeling. Recurrent Neural Networks (RNNs), and later Long Short-Term Memory (LSTM) networks, were designed to better capture sequential dependencies and memory of previous tokens. These models significantly improved the prediction accuracy but still struggled with long-term dependencies and had difficulty handling large-scale

A. Transformer-based Models

The most significant breakthrough in recent years has been the introduction of transformer models, such as BERT (Bidirectional Encoder Representations from Transformers) and GPT (Generative Pretrained Transformer). These models utilize self-attention mechanisms to better capture global dependencies across the entire text sequence, leading to dramatic improvements in next-word prediction tasks. GPT Architecture: GPT, developed by OpenAI, is a causal (unidirectional) language model that is trained using a transformer-based architecture. Its key innovation is its ability to predict the next word in a sequence by attending to all previous words in a left-to-right manner. The GPT series has demonstrated state-of-the-art performance in various NLP tasks, including next-word prediction.

B. Model Architecture

Recurrent Neural Networks (RNN) RNNs are a class of neural networks designed for sequential data. In the context of next-word prediction, RNNs maintain a hidden state that captures information about previous tokens. While effective for short sequences, RNNs face challenges with long-range dependencies and are computationally expensive for large datasets.

LSTM and GRU Networks LSTM (Long Short-Term Memory) networks and Gated Recurrent Units (GRUs) are variants of RNNs designed to mitigate the vanishing gradient problem, which allows them to maintain longer-term depen- dencies.

```
404/404 [==============] - 5s  12ms/step – loss: 0.4826  – accuracy: 0.7847 – val_loss: 0.5099 - Val
l_accuracy: 0.7822
Epoch 296/300
404/404 [==============] – 3s  9ms/step – loss: 0.4527  – accuracy: 0.7995 – val_loss: 0.5149 - Val
_accuracy: 0.7822
Epoch 297/300
404/404 [==============] – 4s  9ms/step – loss: 0.3971 – accuracy: 0.8243 – val_loss: 0.5609 - Val
_accuracy: 0.7624
Epoch 298/300
404/404 [==============] – 3s  9ms/step – loss: 0.4640 – accuracy: 0.7871 – val_loss: 0.5872 - Val
_accuracy: 0.7525
Epoch 299/300
404/404 [==============] – 4s  9ms/step – loss: 0.4335 – accuracy: 0.8317 – val_loss: 0.5533 - Val
_accuracy: 0.7822
Epoch 300/300
404/404 [==============] – 3s  8ms/step – loss: 0.4583 – accuracy: 0.7946 – val_loss: 0.5667 - Val
_accuracy: 0.7822
```

Fig. 1. RESULT

These models improved performance in next-word prediction tasks compared to vanilla RNNs but still suffered from inefficiencies in capturing long-range context.

Transformer Networks Transformers introduced by Vaswani et al. (2017) use a self-attention mechanism that allows the model to weigh the importance of each word in the context of the entire sequence. This is particularly useful for tasks like next-word prediction, as the model can focus on relevant words in the sequence, regardless of their position. The GPT architecture, a generative variant of the transformer, has become the gold standard for next-word prediction.

V. Training and Evaluation Of Model

Dataset and Preprocessing Training a next-word prediction model requires large-scale text corpora, such as books, websites, or social media posts. Common datasets include the Penn Treebank, Wikipedia, and Common Crawl. Preprocessing often involves tokenization, where text is split into words or subwords, and the removal of noise such as special characters.

Loss Functions and Optimization For training models, the objective is usually to minimize a loss function such as cross-entropy, which measures the discrepancy between the predicted probability distribution and the true distribution of the next word. Optimizers like Adam or SGD are used to update model parameters iteratively during training.

Evaluation Metrics Evaluation metrics for next-word prediction models include:

Perplexity: Measures how well the model predicts the next word. Lower perplexity indicates better performance. Accuracy: Measures the percentage of times the predicted word is correct. BLEU and ROUGE: These metrics, typically used for machine translation and summarization, can also be applied to evaluate the quality of predicted sequences in more complex.

Applications Of Next Word Prediction Model

Autocompletion and Text Generation Next-word prediction is widely used in autocompletion systems, where it helps users type faster and with fewer errors. Predictive keyboards, such as those in smartphones, use these models to suggest the next word based on the context.

Conversational AI Next-word prediction is central to building chatbots and virtual assistants, where predicting the next word based on ongoing conversation is essential for maintaining coherent and contextually appropriate responses.

Machine Translation In machine translation, next-word prediction can be used to predict the next word in the target language, given the input sentence in the source language. The more accurate the prediction, the better the quality of translation.

Sentiment Analysis and Content Generation Next-word prediction also plays a key role in content generation systems, including automatic summarization, creative writing, and marketing content. These models generate sentences or paragraphs that are coherent and contextually relevant to the task at hand.

VI. PERFORMANCE ANALYSIS

System evaluation progressed in at least methods depending on the intensity of the usual These often used Analytical / Computational / Statistical / Experimental / or mathematical methods.Outcomes in distinct classes can be compared to extraordinary views Output in distinct categories.N-grams method is inferior to the LSTM method because LSTMs have the memory to examine the setting from the beginning while also paying tribute to the substance corpus. When starting any other business, you should pick one of the current preorganized designs by searching the internet for open-source software.

As a result, you won't have to start from scratch and won't have to be concerned about the association cycle or hyperparameters. Attempting to construct a version using Nietzsche's default textual content file with the goal of anticipating users' sentences after they input forty letters; the version will recognise forty letters and anticipate future letter/words using an LSTM neural network in an effort to apply Tensorflow. exceptional input layers We can observe that regardless of the size of the input layer, the output prediction is accurate to the tune of 54 to 55. The training accuracy for 10 input nodes is around 56 percent, but the checking out accuracy is around 54 percent.The training accuracy for a 20 input node is approximately 56 but the testing accuracy is around 55; for a 30 input node, the training accuracy is around 56 percent,but the testing accuracy is around 55.5 percent, which is identical to 20; and for a 40 input node, the schooling accuracy is around 56 percent. However, the testing accuracy is around fifty-four percent. 9 percent of the population .The Neural network did a fantastic job anticipating the ultimate outcome as you can see in the second case where the model observed the string that comes after "str" by entering 6 check cases and passing 5 in our params (i.E more potent, energy) With a precision of around 56 percent, we're confident in our predictions.

The very final step involves compiling and fitting our version. Here, we're training the model and saving the exceptional weights to nextword1.H5 so that we don't have to retrain it every time we need it and can use our saved model whenever we need it.I have excelled at the schooling

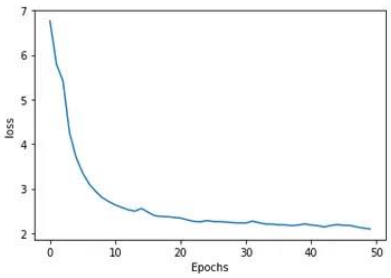

Fig. 2. Accuracy graph

records in this area.You can, however, use both educate and validation statistics to train. We used categorical crossentropy, which calculates the pass-entropy loss between labels and predictions. With a mastery rate of zero.001, Adam is the optimizer we can utilise, and we'll gather our version at the metric loss. The end result is as follows:We may load the tokenizer file that we saved in the pickle format into the prediction notebook.Then we'll load the next word model that we've saved in our directory. This equal tokenizer will be used to tokenize each of the input sentences for which we need to make predictions. Following this, anything could be done.While walking through the predictions, we'll use the try and except statements. We use this statement because we don't want the programme to leave the loop if there is an error in locating the enter sentence. We must execute the script for as long as the customer wants it to run. When the user wants to exit the script, he or she must manually select this option. The software will run as long as the user's objectives are met.If you wish to be offered in the next portion of the item, you can examine this by using the predictions script. Within the next segment, I may provide a link to the GitHub repository. As we can see, the prediction model can best forecast maximum lines.

The stop the script line will exit the application and end the model. The entire software may be terminated if we enter the path "prevent the script." The last phrase of the entered line is used to anticipate the rest of the sentences. We may be considering the very last phrase of each line and attempting to match it with the next word that has the best chance.

VII. MODEL CREATION

We could be building a sequential model. After that, we'll make an embedded layer and specify the input and output sizes. It's crucial to set the duration of the enter to 1, because the prediction could be made with just one phrase, and we'll be able to acquire the answer for that word.Any other layer of the LSTM We can also move it to every other 1000 units in the succeeding layer of LSTM,but there is no need to specify the return collection because it is automatically faked. We'll get around this by using an encrypted 1,000-node layer with a dense layer feature and a com set as activation.

Finally, we send it to the output layer using the specified voice length and softmax activation. When softmax is enabled, we have a lot of options in outputs that are proportional to the size of the voice. Our version structure's whole code is demonstrated here.The three essential callbacks for our model's education will be uploaded. ModelCheckpoint, ReduceLROnPlateau, and Tensorboard are the three most important callbacks. Let's have a look at the functions of each of thesecharactercallbacks.ModelCheckpoint This callback stores the weights of our version once it has been educated.By setting save best only=True, we only save the nice weights of our model. The loss metric will be used to track our education.

ReduceLROnPlateau - This callback is used to reduce the optimizer's studying price after a specific number of epochs. The endurance has been set to three in this case. If the accuracy does not increase after three epochs, we will cut our mastering cost by employing an aspect of zero. 2. Loss is also employed as a monitoring metric in this case.Tensorboard The tensorboard callback is used to visualise graphs, specifically graph plots for accuracy and loss. We'll start by searching on the internet.The following phrase prediction loss graph.We'll save the best styles to the report nextword1.H5 based entirely on the metric loss. This record will be necessary for gaining access to the anticipate function and looking for our next sentence. For the loss to improve, we'll look ahead three epochs. If it does not improve, we will cut the cost of studying. Finally, if desired, we will use the tensorboard function to visualise the graphs and histograms.

A. Predicting the next word

The variable seed text contains the initial input text from which we want to generate the next word predictions. The variable next words indicates the number of words to be predicted. A loop is then executed next words times. Inside the loop, the seed text is tokenized using the tokenizer's texts to sequences.

The model's prediction method is called on the padded token list to obtain the predicted probabilities for each word in the vocabulary. The argmax function is used to determine the index of the word with the highest probability.

The predicted word is obtained by converting the index to the corresponding word using the tokenizer's index word dictionary. The predicted word is then appended to the seed text. This process is repeated for the desired number of next words predictions. Finally, the generated sequence of words is printed as "Next predicted words: [seed text]".

B. Defining the model

The input sequences are mapped to dense vectors of fixed size in the first layer, which is an embedding layer. It requires three arguments: input length (the length of the input sequences less one because we are predicting the next word), 10 (the dimensionality of the embedding space), and total words (the total number of unique words in the vocabulary).

An LSTM (Long Short-Term Memory) layer with 128 units makes up the second layer. Recurrent neural networks (RNNs) of the LSTM variety may recognize long-term dependencies in sequential input. A Dense layer with total words units

and softmax activation make up the third layer. The output probabilities for each word in the vocabulary are generated by this layer. The categorical cross-entropy loss function used in the model is appropriate for multi-class classification applications. Adam is the chosen optimizer, and accuracy is the evaluation metric.

CHALLANGES AND LIMITATIONS

Data and Bias Training data can introduce bias puts. the model, leading to ulraininredictions or offensive outputsMoreover, models trained on large, diverse datasets may not generalize well to specialized domains or languages with less available data.

Computation and Efficiency Modern transformer models like GPT-3 and GPT-4 have billions of parameters, making them computationally expensive to train and deploy. This limits their accessibility and scalability, especially in low-resource environments.

Handling Ambiguity Next-word prediction models may struggle with ambiguity in language, where multiple valid words could follow a given context. Disambiguating the best next word requires a deep understanding of world knowledge and context.

VIII. DEEP LEARNING MODELS

Recurrent Neural Networks (RNNs): Introduce RNNs and their capability to handle sequential data. Discuss how they overcome some limitations of n-gram models by maintaining a memory of previous words in the sequence.

Long Short-Term Memory (LSTM): Explain LSTMs as an improvement over vanilla RNNs, focusing on how they mitigate the vanishing gradient problem, allowing for better modeling of long-term dependencies.

Gated Recurrent Units (GRU): Discuss GRUs, their advantages over LSTMs, and their simplified architecture while maintaining the capability to model sequential data effectively.

IX. FUTURE DIRECTIONS

Multimodal Models The future of next-word prediction models may lie in multimodal approaches, which combine text, images, and speech data. For instance, understanding the context in a conversational AI system could be enhanced by incorporating visual or auditory cues.

Few-shot and Zero-shot Learning Next-generation mod- els will likely focus on reducing the need for large-scale datasets by incorporating few-shot or zero-shot learning, al- lowing models to predict next words with very limited task- specific data.

Ethical and Responsible AI With the increasing de- ployment of predictive models in real-world applications, the ethical implications of next-word prediction, particularly regarding bias and misinformation, need to be addressed. Research into fairness, transparency, and explainability will be crucial in ensuring the responsible use of these models.

X. CONCLUSIONS

Using device learning algorithms such as RNN to comprehend and body texts and stories will help you understand them quickly. Creating lyrics and songs is a crucial field wherein our algorithm can assist stop-users in anticipating the next phrase in songs because the model is taught on a music lyrics data set. As a bonus, we may train the version to review the weights and recognise the centre aspects of paragraphs/sentences so that we can expect favourable results.Paraphrasing is a technique for expressing someone else's ideas in your own words. To paraphrase a source, recreate a section without changing the sense of the original text, so that our algorithms can predict a wider range of phrases from a single sentence and assist users in bordering n sentences. When the gap between the particular situation and the sentence to be expected grows, standard RNNs and other language models become less precise. This is where the LSTM comes in to help with the long-term dependence issue, as it includes memory cells to recall the previous setting.LSTM Neural Net can be examined. Our project for this mission is to teach and strive for a set of rules that are finely tuned for this task, and we are particularly interested in implementing an LSTM to achieve appropriate accuracy, as this task is quite difficult because we must predict the user's future text.We are currently manipulating to understand the problem statement as this is a precise problem. We created a 3d vector layer for input and a second vector layer for output and fed them through to the LSTM layer with 128 hidden layers and controlled to get accuracy to around 56 percent at some point in five epochs. This study explains how the technology predicts and corrects errors.For the metamorphosis dataset, we can expand a massive next phrase prediction. In around 150 epochs, we can significantly reduce the loss. On the available dataset, the next phrase prediction model we constructed is reasonably accurate. The prediction's overall quality is good.

Positive pre-processing steps and version tweaks, on the other hand, can be used to improve the model's prediction.Since March 2006, I've been using the text8 dataset, which is the English Wikipedia sale off. The facts set might be quite extensive, with 16 million words in all. I picked a random subset of statistics with a total of 0.5MM phrases, of which 26k were remarkable phrases, for the purpose of checking out and building a word prediction model. As I will explain later, the answer is no. The intricacy of your version will substantially rise due to the use of unique vocabulary.One of the primary roles in NLP is to replace every word with a phrase vector, which generates a more accurate representation of the phrase's meaning.Please check this blog for additional information on phrase vectors and how they capture semantic meaning. As a result of this design, the RNN can "idea" utilise archaeological statistics to predict the future. However, because pure vanilla RNN has issues with decay and gradient explosion, it is rarely employed. In this example, I employed the LSTM, which employs gates to drift the gradients back in time and solve the extinct gradient problem.In Tensorflow, I created a multi-layer LSTM with 512 devices per layer and

two LSTM. The ultimate 5 phrases are used as input to LSTM, and the target phrase is the following phrase.I used sequence loss as a loss characteristic. For a total of 120 seasons, the model was prepared. To assess educational continuity, I study each teach loss and confusion.

A typical metaphor for grading a language model's overall performance is confusion. The alternative to the standard set check effects for the large range of phrases is what makes things hard. The version number rises to reduce confusion. The model had 35 perplexity after a hundred and twenty epoch instruction.On a few sample tips, I tested the model. The version makes available the pinnacle three words from which the user can choose. See the image below. The model performs effectively because it was trained with a limited vocabulary of 26k words.For the metamorphosis dataset, we can expand a massive next phrase prediction. In around 150 epochs, we can significantly reduce the loss. On the available dataset, the next phrase prediction model we constructed is reasonably accurate. The prediction's overall quality is good.Positive pre-processing procedures and version tweaks can, however, be applied to improve the model's prediction.

REFERENCES

[1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez. A., Kaiser, L., Polosukhin, I. (2017). Attention is all you need. NeurIPS 2017.

[2] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. (2020). Language models are few-shot learners. arXiv preprint arXiv:2005.14165.

[3] Mikolov, T., Chen, K., Corrado, G., Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781

[4] M. D. Zeiler and R. Fergus, "LNCS 8689 - Visualization and Understanding Convolutional Networks, "2014.

[5] A. Dosovitskiy, J. T. Springenberg, M. Tatarchenko, and T. Brox, "Reading Manufacture Seats, Tables and Cars With Convolutional Networks, "IEEE Trans.An analog pattern. March.

[6] J. Portilla and E. P. Simoncelli, quot;Parametric texture model based on integrated calculations ofcomplex wavelet coefficients,quot; Int. J. Computer.

[7] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep-rooted networksin order to study the unsupervised unplanned presentation of program presentations, "2009.

[8] K. Simonyan and A. Zisserman, "Social networks are very deep on a large scale image recognition, "Sep. 2015, Accessed: May 17, 2021.

[9] Bengio, Y., Simard, P., Frasconi, P., 1994.Learning long dependencies with gradient descent is troublesome. IEEE transactions on neural networks five, 157– 166.

[10] Serban, I. V.; Sordoni, A.; Bengio, Y.; Courville, A.; and Pineau, J. 2016. Building end-to-end dialogue mistreatments generative stratified neural network models. In Proceedings of the 30th Conference on Artificial Intelligence. AAAI

[11] Mohd. Majid and Piyush Kumar, Language Modelling: Next word Prediction, 2019.

[12] R. Kneser and H. Ney, "Improved backing-off for n-gram language modeling", Conference on Acoustics speech and Signal Process, pp. 181-184, 1995.