IJSCIA

International Journal of Scientific Advances
ISSN: 2708-7972
Volume: 5 | Issue: 2 | Mar – Apr 2024 Available Online: www.ijscia.com
DOI: 10.51542/ijscia.v5i2.11

Enhancing Armenian Automatic Speech Recognition Performance: A Comprehensive Strategy for Speed, Accuracy, and Linguistic Refinement

Varuzhan H. Baghdasaryan

Armenia, Yerevan, Romanos Melikian 6/1, National Polytechnic University of Armenia

*Corresponding author details: Varuzhan H. Baghdasaryan; varuzh2014@gmail.com

ABSTRACT

This research introduces a comprehensive strategy to enhance the performance of an existing automatic speech recognition (ASR) model, which has been previously documented in published articles. The study sets out to achieve several objectives. Firstly, it concentrates on updating the ASR model by retraining it with new datasets. This involves integrating samples from the latest Common Voice corpus release and data collected independently via the armspeech.com web application. Another key focus lies in optimizing the ASR model for near-real-time processing, intending to improve its speed and efficiency. The proposed adjustments to the model's architecture aim to balance accuracy and processing speed, which is essential for applications requiring prompt speech recognition. Furthermore, the research explores the integration of Transformer models into the post-processing pipeline to introduce punctuation and capitalization into the ASR output. This step not only enhances the linguistic quality of transcriptions but also improves their readability and usability. In tandem with these advancements, the research presents a systematic approach to gathering, annotating, and storing datasets specifically tailored for punctuation and capitalization tasks. The methodology outlines the acquisition and organization of a dataset conducive to training Transformer models for these linguistic tasks. This comprehensive approach, which encompasses dataset enrichment, architectural modifications, and post-processing enhancements, aims to elevate the ASR model's accuracy, speed, and linguistic refinement, with a particular focus on addressing the intricacies of the Armenian language. The research contributes valuable insights into the optimization of ASR systems, tackling both language-specific challenges and broader issues related to linguistic post-processing.

Keywords: Armenian ASR; Armenian automatic speech recognition; Armenian speech-to-text; Armenian speech corpus; Nvidia NeMo; Citrinet; Transformer; DistilBERT; punctuation; capitalization.

INTRODUCTION

Automatic speech recognition (ASR) systems have evolved as essential components in modern technologies, facilitating the conversion of spoken language into written text across various applications with real-time processing. The integration of real-time recognition features alongside improved accuracy in ASR presents challenges, as their synergy depends heavily on factors such as computational capabilities and model design. Generally, opting for a smaller model with fewer parameters is favored to expedite transcription speed. Conversely, maintaining a sufficient number of parameters is crucial for achieving higher accuracy. Additionally, certain ASR systems incorporate punctuation and capitalization as standard elements in their text output.

Previous research has primarily concentrated on three models: Baidu's DeepSpeech, Nvidia's QuartzNet, and Citrinet [1, 2]. Among these, Nvidia Citrinet attained the lowest Word Error Rate (WER) [2]. Therefore, forthcoming research and advancements will primarily concentrate on fine-tuning Nvidia Citrinet to support near-real-time processing abilities. Additionally, there will be efforts to enhance accuracy through the utilization of the newly released Common Voice dataset [3] and data gathered from the armspeech.com web application.

NVIDIA's Citrinet represents a cutting-edge advancement in automatic speech recognition, featuring an end-to-end architecture built on convolutional Connectionist Temporal Classification (CTC) [4]. Leveraging successful ASR models such as QuartzNet and ContextNet, Citrinet's encoder integrates 1D time-channel separable convolutions from QuartzNet with the squeeze-and-excite (SE) mechanism from ContextNet [4]. Additionally, the model employs sub-word encoding via WordPiece tokenization, breaking down words into smaller units known as subwords to represent them as sequences [4]. This innovative approach enables Citrinet to handle previously unseen words and enhances its overall accuracy. The culmination of these design elements enables Citrinet to achieve state-of-the-art accuracy, sometimes surpassing transformer-based counterparts.

This study also confronts the challenge of rectifying deficiencies in ASR system outputs, particularly concerning punctuation and capitalization. Despite their prevalence and advancements, ASR outputs frequently lack proper punctuation and are presented solely in lowercase, thereby impeding readability and contextual accuracy. Punctuation marks serve to organize grammar and effectively convey tone. The precise placement of marks is crucial for conveying accurate meaning within sentences. Similarly, capitalization signifies the start of sentences, proper nouns, and titles, thereby contributing to the visual and structural cohesion of written text. At the core of the methodology of this research lie Transformer models renowned for their ability to capture intricate linguistic relationships. These models hold the promise of significantly enhancing the quality and efficiency of introducing punctuation and capitalization to ASR outputs.

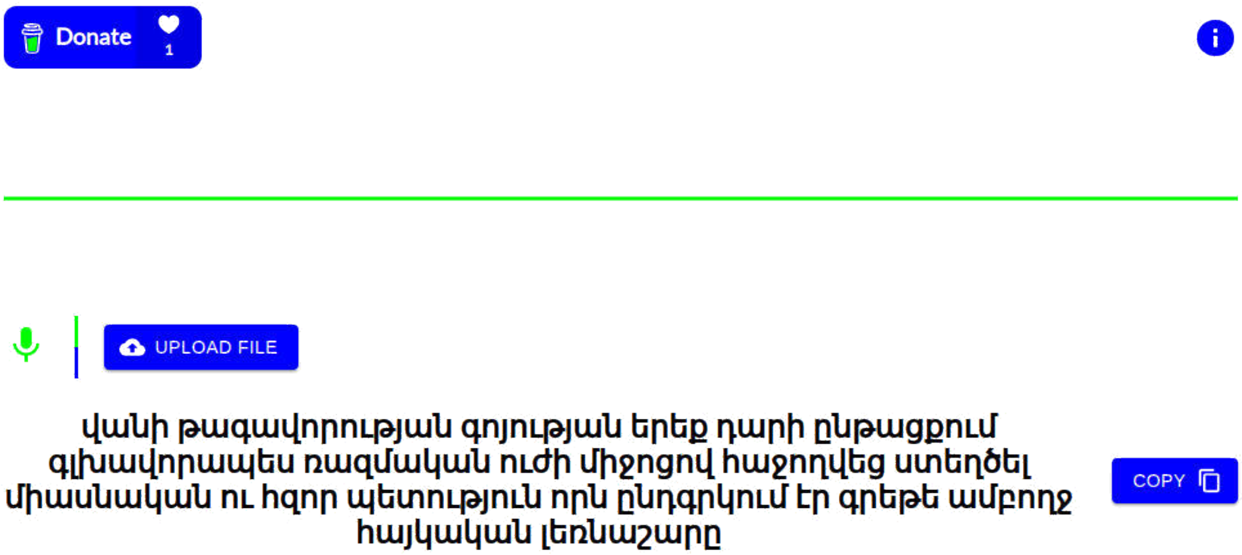
While traditional approaches, governed by rule-based systems, provide guidelines for their proper usage, the advent of natural language processing (NLP) has ushered in Transformer models like BERT, which have revolutionized the understanding and generation of language. Unlike traditional sequential models, BERT excels in bidirectional language understanding, meticulously considering the contextual nuances of each word within its surrounding context [5]. Pre-trained on extensive corpora, BERT epitomizes the advancement in NLP, offering unparalleled capabilities in language processing and paving the way for more accurate and contextually-aware language processing systems [5].

The structured dataset essential for training Transformer models for tasks like punctuation and capitalization must adhere to specific guidelines. Meeting the demands of these tasks requires a large quantity of data, with the Armenian internet being the primary reservoir for such resources.

Overall, this research endeavors to mitigate this limitation by focusing on enhancing the speed and accuracy of ASR models while refining textual outputs. The research underscores the importance of integrating appropriate punctuation and capitalization, recognizing their significance in improving usability and comprehension.

SPEECH RECOGNITION DATASET

Previous research primarily focused on creating an Armenian universal speech corpora [6, 7] and conducting experiments using different ASR models tailored for the Armenian language [1, 2]. Among these models, the Nvidia Citrinet demonstrated the highest accuracy, achieving a WER of 19.41% without an additional n-gram language model (LM) [2]. Subsequently, launched armspeech.com, a website with a user-friendly graphical user interface (see Figure 1).



վանի թագավորության գոյության երեք դարի ընթացքում գլխավորապես ռազմական ուժի միջոցով հաջողվեց ստեղծել միասնական ու հզոր պետություն որն ընդգրկում էր գրեթե ամբողջ հայկական լեռնաշարը

FIGURE 1: The view of the armspeech.com website.

The main goal of armspeech.com was to assess the performance of the Citrinet model in real-world scenarios, identify areas for improvement in accuracy and speed, and collect additional data to refine the model further. The website's backend employed an acoustic Citrinet model combined with a KenLM language model for recognizing Armenian speech. However, the results lacked capitalization and punctuation. Notably, the model was compact, and the server relied on CPU processing, leading to slightly longer processing times.

The armspeech.com web application significantly contributed to the expansion of the dataset. Data submitted through the web application were anonymized and stored for future model enhancements. Over three months, armspeech.com gained popularity among the Armenian-speaking community, accumulating 21.53 hours of speech data from both native and non-native speakers, covering various everyday topics. The subsequent step involved manually verifying transcriptions to correct any inaccuracies.

Furthermore, the Armenian resources within the Common Voice project saw a size increase of approximately 23 validated hours [3]. However, some of the validated samples from Common Voice contained characters, typically numbers and symbols such as “&”, “@”, and “$”. These samples were removed using an additional Python script.

By amalgamating the datasets utilized in earlier research endeavors (namely, the Armspeech corpus, speech corpus of Armenian question-answer dialogues, and Google's FLEURS, together amounting to 34.84 hours) [1, 2, 6, 7], the ultimate dataset size attained a total of 79.37 hours of data. The structure of the dataset for training ASR models has been deeply researched in previous articles [6, 7].

Analysis of the statistics gathered from armspeech.com indicates that there is a requirement for enhancements in both accuracy and overall processing time to achieve improved performance. Consequently, before delving into fine-tuning the model with the newly collected dataset, it became apparent that structural modifications were imperative.

SPEECH RECOGNITION MODEL

There has been limited research on adjusting the parameters and structure of Citrinet to improve accuracy and inference time for more complex languages.

For example, Xianchao Wu investigated the adaptation of Citrinet for Japanese [8]. The main focus was on integrating multi-head attentions into the convolution module of Citrinet blocks to reduce the number of CNN layers while keeping SE and residual modules unchanged [8]. This approach involved trimming eight convolution layers in each attention-enhanced Citrinet block to facilitate quicker training [8]. Additionally, the original 23 Jasper blocks were condensed to eight blocks, resulting in a significant reduction in model size [8]. This strategy ensured that attention-enhanced Citrinet achieved similar inference times for extended speech sequences ranging from the 20s to the 100s layers [8]. Evaluations conducted on the Japanese CSJ-500-hour dataset demonstrated that attention-Citrinet achieved faster convergence and lower character error rates with fewer block layers compared to Citrinet, showcasing an 80% reduction in training time [8].

Numerous avenues exist for tailoring Citrinet to address specific requirements, including achieving both fast performance and higher accuracy in automatic speech recognition. Initially, adjustments can be made in terms of the model's structural components, such as the number of layers and filters. By fine-tuning the convolutional layers and filter quantities, one can regulate the model's complexity and capacity.

Another facet to consider is the choice of decoder type within Citrinet. The default decoder, CTC, typically suits a broad range of ASR tasks [4, 9]. However, for endeavors necessitating precise alignment between audio and text, like keyword spotting or speaker diarization, RNN-T (Recurrent Neural Network Transducer) serves as an alternative decoder type [4, 9].

Vocabulary size represents another pivotal parameter ripe for customization. By adjusting the vocabulary of subword units, one can strike a balance between accuracy and efficiency. Larger vocabularies enable capturing finer linguistic nuances, whereas smaller ones enhance speed and reduce memory consumption [4, 9].

Additionally, an alternative approach to reduce the processing time of the current Armenian ASR model involves eliminating the supplementary statistical language model. While this model enhances recognition accuracy, its usage also extends processing duration and necessitates the integration of additional libraries and scripts into the ASR model's operational environment [1, 2].

Augmenting the training data can significantly enhance model robustness in real-world scenarios. Techniques such as noise injection, speed and pitch variation, and reverberation broaden the diversity of training data, thereby bolstering model performance.

Model quantization offers an avenue to optimize accuracy by representing model weights and activations with lower precision, thus diminishing memory footprint and inference time.

Hyperparameters fine-tuning constitutes another strategy to refine ASR accuracy. Variations in learning rate, batch size, optimizer, and weight decay can be explored to optimize model convergence and accuracy.

PUNCTUATION AND CAPITALIZATION DATASET

The datasets utilized in training models to punctuate and capitalize text like ASR output are primarily of two types: parallel and unlabeled.

Parallel corpora comprise pairs of texts, wherein one version is fully punctuated and capitalized, while the other is presented in raw form without these linguistic elements. These datasets play a crucial role in training models to anticipate punctuation and capitalization based on contextual cues and language patterns. Examples of parallel corpora include translated text transcripts and books containing both original and plain text versions.

Unlabeled corpora containing punctuation encompass vast collections of text characterized by accurate punctuation and capitalization. Models trained on such datasets implicitly learn patterns without explicit labels, thereby improving their capacity to comprehend and produce properly punctuated and capitalized text. Examples of unlabeled corpora with punctuation include news articles and Wikipedia entries.

This research employs parallel corpora with sentence-level annotation, where each sample represents a single sentence complete with punctuation and capitalization. The final model preparation leverages the NeMo toolkit provided by Nvidia, which serves as a comprehensive generative AI framework for ASR, speech synthesis, punctuation, capitalization, and other tasks [9].

In the realm of the Armenian language, punctuation presents intricate challenges that hinder the preparation of datasets and the training of punctuation models. Notably, Armenian punctuation diverges from conventional norms, as it employs the "՞" symbol instead of the commonly recognized "?" for marking question sentences. Remarkably, Armenian stands alone among ancient languages in its placement of the question mark directly on the stressed vowel (typically the final vowel) of the interrogative word within the sentence, contrasting with English where it traditionally concludes the sentence. Furthermore, Armenian allows for the occurrence of multiple question marks within a single sentence.

The aforementioned differences between Armenian and other widely spoken languages hinder the use of the NeMo toolkit. The technique utilizing the NVIDIA NeMo toolkit for punctuation and capitalization prediction operates based on assigning only one punctuation mark per word, in addition to tagging capitalization [9]. Consequently, during dataset preprocessing, a decision was made to substitute all Armenian question marks with "?" marks, relocating them to the end of the respective word. This implies that the model will be trained to recognize questions and add a "?" after the respective question words. However, during testing, an additional script will be needed to replace the "?" with its Armenian equivalent and position it correctly over the stressed vowel of the word, taking into account any exceptional cases.

Complicating matters further are instances where a word contains two punctuation marks: a question mark and a comma after the word.

Given the current model's limitation of predicting only one punctuation mark per word, it was deemed necessary to replace the combination of the Armenian question mark and comma on the same word with a distinct marker, denoted as "*". Once more, following the prediction phase, an extra script will be required to substitute the "*" marker with the Armenian question mark, ensuring its accurate placement within the word, and to append a comma after the word.

The NeMo toolkit punctuation and capitalization model is capable of processing any dataset as long as it adheres to the specified format below [9].

Before data pre-processing to match the model's expectations, it is imperative to divide the data into training and validation sets with an 80-20% ratio. Each line within these sets represents a single sentence, aligning with the task's objective of punctuating one complete sentence. The training and validation sets must be structured as a combination of two files: text.txt and labels.txt [9].

In the text.txt file, each line comprises text sequences, with lowercase words separated by spaces: [WORD] [SPACE] [WORD] [SPACE] [WORD] [9].

On the other hand, the labels.txt file should adhere to the following format: [LABEL] [SPACE] [LABEL] [SPACE] [LABEL], containing corresponding labels for each word in text.txt [9]. Each label in labels.txt is composed of two symbols:

- the first symbol of the label denotes the punctuation mark that should succeed the word (where "O" signifies no punctuation is needed) [9],
- the second symbol indicates whether a word should be capitalized or not (where "U" signifies the word should be uppercased, and "O" denotes no capitalization is necessary) [9].

An illustrative example of the aforementioned can be found in Table 1.

TABLE 1: An example of a dataset sample.

Original sentence	text.txt	labels.txt
Գիտե՞ք, թե ով է նա	գիտեք թե ով է նա	*U OO OO OO OO
Ողջույն, ես Արմենն եմ	ողջույն ես արմենն եմ	,U OO OU OO

The punctuation marks taken into account include the comma (","), question mark ("?"), and a combination of question mark and comma ("*"). All other punctuation marks were excluded from the dataset.

The full stop mark (represented as ":" in Armenian) was also excluded from the dataset due to the task's objective of punctuating and capitalizing sentences individually, treating each output of the ASR model as one complete sentence. The comprehensive list of all potential labels utilized in this research is in Table 2.

TABLE 2: The list of all possible labels.

Label	Example
OO	տարբեր
OU	Տարբեր
,O	տարբեր,
,U	Տարբեր,
?O	տարբե՞ր
?U	Տարբե՞ր
*O	տարբե՞ր,
*U	Տարբե՞ր,
<blank space>	<blank space>

Certainly, to avoid manual annotation (time-consuming) the essential data for training the BERT model was gathered from various Armenian sources on the internet. This encompassed news articles covering topics such as politics, sports, history, style, and more. Additionally, the Tatoeba Armenian resources were utilized.

A Python script was used to scrape and organize the dataset. Articles from Armenian news websites were processed in batches of 20. The text body of each article was extracted, and the text was segmented into sentences. These sentences then underwent a series of processing steps:

- sentences entirely in uppercase were skipped,
- sentences containing unclosed quotes were skipped,
- sentences with characters outside of Armenian uppercase and lowercase letters, numbers (0-9), and specific punctuation marks (comma, Armenian full stop, dash, Armenian quotation, Armenian question, emphasis, and exclamation) were skipped as well.

Furthermore, duplicated sentences were removed, resulting in a total collection of 11.171.132 distinct sentences.

The subsequent stage involved rendering the collected sentences. During this phase, all punctuation marks were removed from the sentences except for those necessary to reconstruct the BERT model within the sentence. Any dash was substituted with a space, and Armenian question marks were changed to "?", and positioned at the end of the words. Additionally, any occurrences of question marks and commas in the same word were replaced with an asterisk at the word's end. Finally, any remaining punctuation marks were simply removed.

PUNCTUATION AND CAPITALIZATION MODEL

BERT (Bidirectional Encoder Representations from Transformers) is a type of model based on the Transformer architecture, specifically designed for NLP tasks and has been groundbreaking in NLP due to its ability to capture bidirectional context, allowing it to understand the meaning of words in a sentence by considering the context in which they appear [5]. BERT consists of multiple layers of Transformers, each composed of self-attention mechanisms and feedforward neural networks [5].

BERT utilizes a pre-training and fine-tuning approach [5]. During pre-training, the model is trained on a large corpus of text using unsupervised learning objectives such as masked language modeling (MLM) and next-sentence prediction (NSP) [5]. In MLM, certain words in the input sequence are masked, and the model is trained to predict these masked words based on the surrounding context [5]. NSP involves training the model to predict whether a sentence follows another sentence in the corpus [5].

BERT can be used for text punctuation and capitalization by fine-tuning the model on a specific task related to punctuation and capitalization prediction. By providing input sequences with punctuations and capitalizations removed, and training the model to predict the correct punctuation marks and capitalization, BERT can learn to accurately restore these elements to the text. BERT tokenizes input text into smaller units called tokens [5]. Special tokens, such as [CLS] (classification) and [SEP] (separator), help the model understand the structure of the input [5]. The model represents each token with an embedding that captures both its meaning and context and can analyze surrounding words and context to determine where commas, periods, and other marks belong. Similar to punctuation restoration, BERT can identify and correct capitalization errors, ensuring proper names and titles receive their due.

BERT boasts impressive accuracy compared to traditional rule-based methods, particularly in complex or ambiguous cases. Trained on diverse data, BERT can adapt to various writing styles and domains, making it a one-model-fits-many solution. Pre-trained BERT models eliminate the need for extensive training from scratch, saving time and computational resources.

DistilBERT is a distilled (smaller and faster) version of BERT developed by Hugging Face in 2019 [10]. It aims to retain the performance of BERT while being more efficient in terms of computational resources and memory. DistilBERT has fewer layers and fewer attention heads compared to the original BERT model, resulting in a smaller model size [10]. It uses a technique called parameter distillation, where knowledge from the original BERT model is transferred to a smaller model during training [10]. Due to its reduced size, DistilBERT is faster and more resource-efficient than BERT, making it more suitable for deployment in resource-constrained environments such as mobile devices or low-power servers [10]. Although DistilBERT sacrifices some performance compared to BERT, it still achieves competitive results on various NLP tasks while being more efficient.

The Nvidia NeMo framework offers the capability to fine-tune any pre-trained model [9]. In this case, transfer learning will be performed using a pre-trained model called “distilbert-base-uncased-finetuned-sst-2-english”, which is a fine-tuned version of “DistilBERT-base-uncased” on the SST-2 dataset [10]. According to the paper, this fine-tuned model achieves an accuracy of 91.3% on the development set [10]. For comparison, the “bert-base-uncased” version of BERT reaches an accuracy of 92.7% [10].

The NeMo toolkit offers two variations of punctuation and capitalization models [9]:

- lexical-only model,
- lexical-audio model.

At times, punctuation and capitalization may not be recoverable solely from text. When utilizing a lexical-audio model, incorporating audio data can enhance the model's accuracy. However, this necessitates extra datasets and overcoming various challenges. Additionally, it inevitably leads to longer processing times for the model. Therefore, within the scope of this research, a lexical-only model will be used.

TRAINING AND RESULTS

The optimal configuration for Citrinet hinges upon the specific use case and available resources. In this research, the aim is to achieve higher accuracy with the capability of fast inference, akin to near-real-time processing. While default configurations exist, research endeavors to tailor them to meet specific needs and explore diverse architectural choices. The NeMo framework furnishes tools facilitating facile modification and experimentation with model architectures, underscoring the value of testing different configurations to identify the most suitable combination [9].

Recent studies leveraging Citrinet, despite dataset sizes not exceeding 100 hours, have demonstrated enhanced accuracy.

To optimize Citrinet-256, the initial step involved fine-tuning the number of Jasper blocks, utilizing 23 (default) and 16 blocks, respectively. The decoder remained unchanged, leveraging the default CTC decoder, which exhibited superior accuracy [4, 9].

Pretrained English models boast vocabulary sizes of 1024 for Citrinet variants [4, 9], aligning with the current research's reliance on transfer learning from English pre-trained models. Consequently, the vocabulary size remains at 1024, with pre-trained weights sourced from English models. In line with the previous study, will be applied SpecAugment and SpecCutout with identical parameters for data augmentation, which enhances training data diversity effectively [2]. For weight and activation precision, a 16-bit floating-point representation will be utilized throughout the experiment.

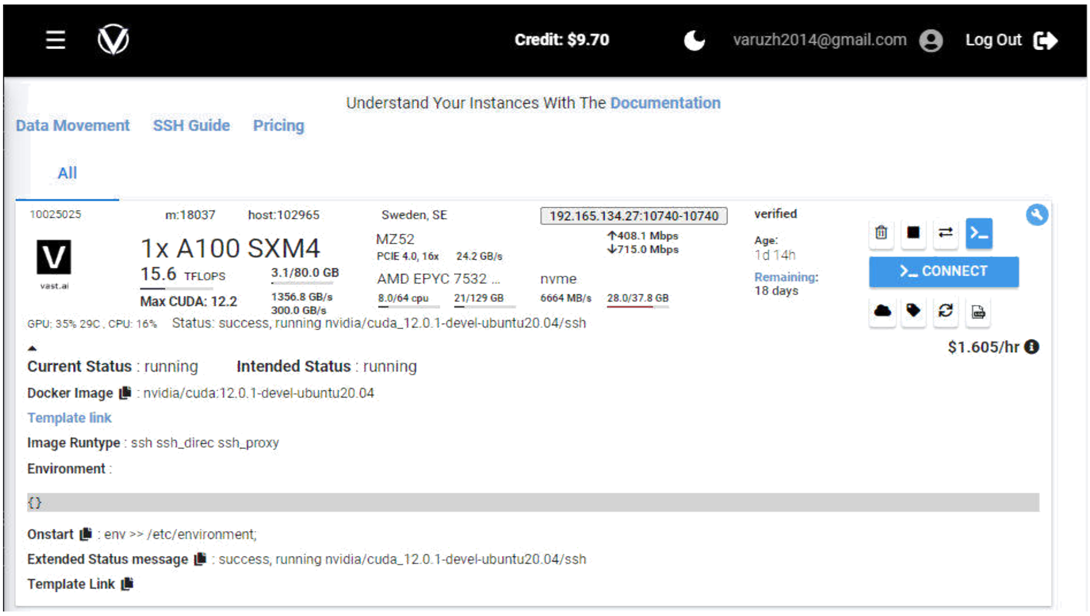
The pre-trained English model, featuring 256 filters of each convolution layer in each block and a size of 40.14 MB, underwent training with consistent hyperparameters: a learning rate of 0.0001, 100 epochs, a batch size of 4, weight decay set at 0.001, and utilizing the NovoGrad optimization algorithm.
In previous articles, various ASR models (Baidu’s DeepSpeech, Nvidia QuartzNet, and Citrinet) incorporated additional n-gram language models to enhance accuracy, albeit at the cost of transcription speed [1, 2]. However, for this study, no additional language model was employed to prioritize faster inference time.

The ASR dataset utilized comprised a total duration of 83.37 hours, divided into training (66.7 hours) and validation (16.67 hours) sets, maintaining an 80-20% ratio.

Experiments for training and evaluating the acoustic models were conducted on a machine running a 64-bit Linux OS (Ubuntu 22.04 LTS) with an Intel Core i7-9700 CPU clocked at 3 GHz. The machine was equipped with two NVIDIA GPUs: NVIDIA GeForce GTX 1660 TUF and NVIDIA GeForce GTX 1650, totaling 10GB of memory.

After undergoing 100 epochs of training, Citrinet equipped with 16 blocks of Jasper achieves a WER of 18.2%, all without the use of any external n-gram language model. Despite attempts to lower the number of Jasper blocks for the Armenian language, it did not significantly enhance accuracy compared to the model employing 23 Jasper blocks. However, it did lead to a noticeable reduction in processing time by 26%.

However, when the default number of Jasper blocks (23) was maintained, the model performed optimally, achieving a WER of 13.4% after 100 epochs without utilizing an external language model. This result represents an improvement compared to the WER of 19.41% reported in previous articles [1, 2].



Credit: $9.70 varuzh2014@gmail.com Log Out

Understand Your Instances With The Documentation

Data Movement SSH Guide Pricing

All

10025025 m:18037 host:102965 Sweden, SE 192.165.134.27:10740-10740 verified

1x A100 SXM4

15.6 TFLOPS 3.1/80.0 GB

Max CUDA: 12.2 1356.8 GB/s 300.0 GB/s

MZ52 PCIE 4.0, 16x 24.2 GB/s

AMD EPYC 7532 ... 8.0/64 cpu 21/129 GB

nvme 6664 MB/s 28.0/37.8 GB

↑408.1 Mbps ↓715.0 Mbps

Age: 1d 14h

Remaining: 18 days

GPU: 35% 29C , CPU: 16% Status: success, running nvidia/cuda_12.0.1-devel-ubuntu20.04/ssh

$1.605/hr

Current Status : running Intended Status : running

Docker Image : nvidia/cuda:12.0.1-devel-ubuntu20.04

Template link

Image Runtype : ssh ssh_direc ssh_proxy

Environment :

{}

Onstart : env >> /etc/environment;

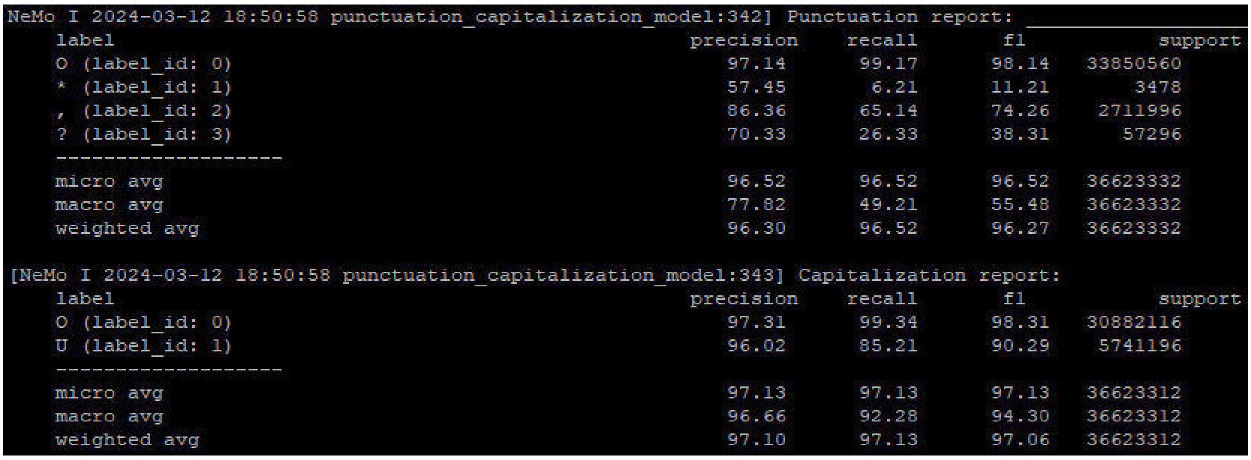
Extended Status message : success, running nvidia/cuda_12.0.1-devel-ubuntu20.04/ssh

Template Link

FIGURE 2: Details of Nvidia A100 SXM4 GPU used for punctuation and capitalization model training.

The punctuation and capitalization model training opted for 16-bit floating-point precision in weight and activation. The total size of the model is 265.4 MB. Adam optimization algorithm was employed for training without weight decay. The model comprises 66.4 million trainable parameters, with 3.1 thousand allocated for the punctuation token-level classifier and 1.5 thousand for the capitalization token-level classifier. The learning rate is set to 0.0001, with a batch size of 2048 tokens and a maximum sequence length of 128.

Due to the substantial size of the dataset (11.171.132 samples) for punctuation model training, the computational resources available for ASR model training were insufficient. Instead, an online machine was rented, equipped with one Nvidia A100 SXM4 GPU boasting 80 GB of memory (see Figure 2). Training duration amounted to 1 day and 14 hours, culminating in 10 epochs with a validation loss of 0.12573. Comprehensive training details, including precision, recall, and F1 scores, are provided in Figure 3.



```
NeMo I 2024-03-12 18:50:58 punctuation_capitalization_model:342] Punctuation report:
    label                                                precision    recall       f1       support
    O (label_id: 0)                                         97.14      99.17      98.14   33850560
    * (label_id: 1)                                         57.45       6.21      11.21       3478
    , (label_id: 2)                                         86.36      65.14      74.26    2711996
    ? (label_id: 3)                                         70.33      26.33      38.31      57296
    -------------------
    micro avg                                               96.52      96.52      96.52   36623332
    macro avg                                               77.82      49.21      55.48   36623332
    weighted avg                                            96.30      96.52      96.27   36623332

[NeMo I 2024-03-12 18:50:58 punctuation_capitalization_model:343] Capitalization report:
    label                                                precision    recall       f1       support
    O (label_id: 0)                                         97.31      99.34      98.31   30882116
    U (label_id: 1)                                         96.02      85.21      90.29    5741196
    -------------------
    micro avg                                               97.13      97.13      97.13   36623312
    macro avg                                               96.66      92.28      94.30   36623312
    weighted avg                                            97.10      97.13      97.06   36623312
```

FIGURE 3: Punctuation and capitalization model training results.

CONCLUSION

Within this research, three primary objectives were delineated: firstly, to optimize the performance of an existing automatic speech recognition model through retraining it with a novel dataset; secondly, to enable swift (near-real-time) transcription capabilities; and thirdly, to augment the ASR model's output by incorporating features for punctuation and capitalization, particularly for unpunctuated and lowercase transcriptions.

The dataset utilized for this research was anonymously collected through the armspeech.com web application, supplemented by the latest release from the Common Voice project. Using the default number of Jasper blocks, the model accomplished a WER of 13.4%, notably surpassing the previous model's performance as detailed in the preceding article, which attained a WER of 19.41%.

Furthermore, the elimination of an external language model, which had been consuming significant time due to its performance, resulted in a decrease in speech recognition time.

To facilitate near-real-time processing, modifications were made to the original structure of the Citrinet ASR model. Specifically, the number of Jasper blocks in the architecture was reduced from 23 to 16, rendering the model more lightweight and decreasing both training and recognizing times. Despite this modification leading to a slight rise in the WER to 18.2%, speech recognition time, on the other hand, observed an average decrease of 26%.

Furthermore, a punctuation and capitalization model, based on the Transformer Distilbert architecture, was trained to refine the ASR output, ensuring readability, usability, and grammatical correctness. Trained on a comprehensive dataset sourced from Armenian language sources on the web, this model adeptly capitalized proper names and punctuated text according to Armenian grammar conventions, including unique attributes such as the placement of the Armenian question mark directly on the stressed vowel within interrogative words, allowance for multiple question marks within a single sentence, and instances of words containing both a question mark and a comma.

It's noteworthy that sentence-ending punctuation marks, such as the Armenian full stop mark (“:”), were not utilized in this model, as the task focused on punctuating sentences as a whole. Despite this, the model demonstrated commendable accuracy, precision, and F1 score metrics.

Lastly, owing to the computational efficiency of the ASR model (with 256 filters of the convolution layers in each block) and the lightweight nature of the punctuation and capitalization model, this system exhibits the capability for near-real-time processing in the Armenian language, particularly under conditions of robust computational resources.

REFERENCES

[1] V. H. Baghdasaryan, “Armenian Speech Recognition System: Acoustic and Language Models”, International Journal of Scientific Advances (IJSCIA), Volume 3| Issue 5: Sep-Oct 2022, Pages 719-724, URL: https://www.ijscia.com/wp-content/uploads/2022/09/Volume3-Issue5-Sep-Oct-No.339-719-724.pdf.

[2] V. H. Baghdasaryan, “Exploring Armenian Speech Recognition: A Comparative Analysis of ASR Models – Assessing DeepSpeech, Nvidia NeMo QuartzNet, and Citrinet on Varied Armenian Speech Corpora”, CSIT Conference 2023, Yerevan, Armenia, September 25 – 30.

[3] Ardila, Rosana, Megan Branson, Kelly Davis, Michael Henretty, Michael Kohler, Josh Meyer, Reuben Morais, Lindsay Saunders, Francis M. Tyers, and Gregor Weber, “Common Voice: A Massively-Multilingual Speech Corpus”, Proceedings of the 12th Language Resources and Evaluation Conference, May 2020, pages 4218-4222.

[4] Somshubra Majumdar, Jagadeesh Balam, Oleksii Hrinchuk, Vitaly Lavrukhin, Vahid Noroozi, Boris Ginsburg, “Citrinet: Closing the Gap between Non-Autoregressive and Autoregressive End-to-End Models for Automatic Speech Recognition”, NVIDIA, USA, arXiv:2104.01721 [eess.AS], 5 Apr 2021, https://doi.org/10.48550/arXiv.2104.01721.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”, arXiv:1810.04805 [cs.CL], 24 May 2019, https://doi.org/10.48550/arXiv.1810.04805.

[6] V. H. Baghdasaryan, “ArmSpeech: Armenian Spoken Language Corpus”, International Journal of Scientific Advances (IJSCIA), Volume 3| Issue 3: May-Jun 2022, Pages 454-459, URL: https://www.ijscia.com/wp-content/uploads/2022/06/Volume3-Issue3-May-Jun-No.283-454-459.pdf.

[7] V. H. Baghdasaryan, “Extended ArmSpeech: Armenian Spoken Language Corpus”, International Journal of Scientific Advances (IJSCIA), Volume 3| Issue 4: Jul-Aug 2022, Pages 573-576, URL: https://www.ijscia.com/wp-content/uploads/2022/09/Volume3-Issue4-Jul-Aug-No.309-573-576.pdf.

[8] Xianchao Wu, “Attention Enhanced Citrinet for Speech Recognition”, arXiv:2209.00261 [cs.CL], 1 Sep 2022, https://doi.org/10.48550/arXiv.2209.00261.

[9] Harper, E., Majumdar, S., Kuchaiev, O., Jason, L., Zhang, Y., Bakhturina, E., Noroozi, V., Subramanian, S., Nithin, K., Jocelyn, H., Jia, F., Balam, J., Yang, X., Livne, M., Dong, Y., Naren, S., & Ginsburg, B., “NeMo: a toolkit for Conversational AI and Large Language Models”, [Computer software], URL: https://github.com/NVIDIA/NeMo.

[10] Victor Sanh, Lysandre Debut, Julien Chaumond, Thomas Wolf, “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter”, arXiv:1910.01108 [cs.CL], 1 Mar 2020, https://doi.org/10.48550/arXiv.1910.01108.