DATA ANALYTICS USING DEEP LEARNING

GT 8803 // FALL 2019 // JOY ARULRAJ

WRITING TIPS

CREATING THE NEXT®

ANALYSIS

ANALYSIS

- Problem Description
- Significance
- Novelty
- Relevance
- Validity
- Contribution

PROBLEM DESCRIPTION

- What is the problem being considered?
- Is it clearly stated?
- What are the important issues?
- Early in the report, clarify what has been accomplished?
  - For example, if this is a system description, has the system been implemented or is this just a design?

SIGNIFICANCE

- Is the goal of this paper significant?
- Is the problem real?
- Is there any reason to care about the results of this paper, assuming for the moment that they are correct?
- Is the problem major, minor, trivial or non-existent?

RELEVANCE

- Is the problem now obsolete, such as reliability studies for vacuum tube mainframe computers?
- Is the problem so specific or so applied as to have no general applicability and thus not be worth wide publication?

NOVELTY

- Is the problem, goal, or intended result new?
- Has it been built before?
- Has it been solved before?
- Is this a trivial variation on or extension of previous results?
- Is the author aware of related and previous work, both recent and old?

VALIDITY

- Is the method of approach valid?
- What are the assumptions? How realistic are they?
- If they aren't realistic, does it matter?
- How sensitive are the results to the assumptions?

CONTRIBUTION

- What did you, or what should the reader, learn from this paper?
- If you didn't learn anything, and/or if the intended reader won't learn anything, the paper is not publishable

WRITING TIPS

WRITING TIPS

- Bulleted Lists
- Weasel Words
- Salt & Pepper Words
- Beholder Words
- Lazy Words
- Adverbs
- Tools

WRITING TIP #1: BULLETED LIST

- Don’t write verbose paragraphs
  - Use bulleted lists

WRITING TIP #2: WEASEL WORDS

- Weasel words--phrases or words that sound good without conveying information--obscure precision.

WRITING TIP #2: SALT & PEPPER WORDS

- New grad students sprinkle in salt and pepper words for seasoning. These words look and feel like technical words, but convey nothing.
- Examples: *various*, *a number of*, *fairly*, and *quite*.
- Sentences that cut these words out become stronger.

WRITING TIP #2: SALT & PEPPER WORDS

- Bad: It is quite difficult to find untainted samples.
  - Better: It is difficult to find untainted samples.
- Bad: We used various methods to isolate four samples.
  - Better: We isolated four samples.

WRITING TIP #3: BEHOLDER WORDS

- Beholder words are those whose meaning is a function of the reader
- Example: *interestingly*, *surprisingly*, *remarkably*, or *clearly*.
- Peer reviewers don't like judgments drawn for them.

WRITING TIP #3: BEHOLDER WORDS

- Bad: False positives were surprisingly low.
- Better: To our surprise, false positives were low.
- Good: To our surprise, false positives were low (3%).

WRITING TIP #4: LAZY WORDS

- Students insert lazy words in order to avoid making a quantitative characterization. They give the impression that the author has not yet conducted said characterization.
- These words make the science feel unfirm and unfinished.

WRITING TIP #4: LAZY WORDS

- The two worst offenders in this category are the words *very* and *extremely*. These two adverbs are never excusable in technical writing. Never.
- Other offenders include *several*, *exceedingly*, *many*, *most*, *few*, *vast*.

WRITING TIP #4: LAZY WORDS

- Bad: There is very close match between the two semantics.
- Better: There is a close match between the two semantics.

WRITING TIP #5: ADVERBS

- In technical writing, adverbs tend to come off as weasel words.
- I'd even go so far as to say that the removal of all adverbs from any technical writing would be a net positive for my newest graduate students. (That is, new graduate students weaken a sentence when they insert adverbs more frequently than they strengthen it.)

WRITING TIP #5: ADVERBS

- Bad: We offer a completely different formulation of CFA.
- Better: We offer a different formulation of CFA.

WRITING TIP #6: LEVERAGE TOOLS

- Tools
  - https://github.com/jarulraj/checker
  - http://matt.might.net/articles/shell-scripts-for-passive-voice-weasel-words-duplicates/

WRITING TIP #7: STRENGHTS

- Bad: Open sourcing the algorithm.
- Bad: Easy to implement the algorithm using libraries.
- Bad: Does a good job of describing optimizations at each step.
- Bad: Paper also does a few real world tests.
- Bad: Paper provides theoretical guarantees about the bounds.

WRITING TIP #7: STRENGHTS

- Good: Detection of new, low-magnitude earthquakes that were previously not detected.
- Good: Accelerates query processing by 100x.
- Good: The authors consider human attributes such as limited cognitive load and short attention span.

WRITING TIP #7: STRENGHTS

- Bad: Since the authors collaborated with seismologists for their research, their domain knowledge is well represented.
- Better: They introduce the following domain-specific optimizations: X, Y, Z.

EXAMPLES

In certain areas, low bandwidth networks exist, such as 3G cellular network. It dramatically affects the data communication speed between the edge and the cloud. In order to improve the performance for the video analytics system with bad networking condition, we propose the customized compression algorithm on top of compressed sparse row format. The proposed compression technique specifically targets to improve the data transfer speed under the edge and the cloud quo placement. Since the data transfer usually happens after the edge finishes some layers' computations, the compression technique is typically applied to the intermediate outputs generated from the neural network.

In regions with lower bandwidth networks (*e.g.*, 3G cellular network [17]), the communication cost between the edge and the cloud devices is high. For instance, it takes 80 s to transfer 1 MB of data in a 3G network. Reducing the amount of data transferred is critical to maximize the throughput of System X in such settings. To accomplish this, we propose a novel compression scheme based on the CSR representation [1, 3]. This scheme is specifically tailored for the output tensor of an intermediate layer of the model that is sent by the edge device to the cloud device. These tensors share the following properties:

We first attempt to understand the property of the intermediate outputs by profiling the unique value and non-zero value of each layer in the multi-stage neural network. We observe that ReLU layer [5] and Max Pooling layer are usually used in a DNN. ReLU often adds sparsity to the intermediate matrices because it changes all negative activations to zero. Max Pooling downsamples the matrices size. The Table 6 shows the according outputs value sparsity and uniqueness properties for each layer building block. Some layer units end with ReLU but some end with Max Pooling layer. For example, the first row profiles the intermediate value generated after the first layer unit, which ends with ReLU layer. The intermediate matrix has 25.99% unique entries and about 60.33% entries in the entire matrix is zero value.

From Table 6, we observe that for intermediate value in neural networks, more than 60% is zero value. The output generated after Max Pooling usually has smaller sparsity, but more unique non-zero values. For all intermediate output matrices, non-zero redundant value take very small portion in the entire matrix (the sum of sparsity and unique percentage is very close to 100%). Due to that, we can achieve reasonably good compression ratio by simply using the compressed sparse row format. It greatly reduces space used by zero value. Other general compression algorithms will not help because the matrix has very few non-zero redundant value. In later experiments, we want to use CSR format as the baseline of a new compression technique. We want

❶ **Sparsity**: The ReLU layer converts all the negative values in the input tensor to zero. This increases the sparsity of the matrix. For example, Table 6 presents the sparsity of the tensors produced by the different layers of the MultiNN VGG-16 model on the Flower-102 dataset [35]. The average sparsity of the tensors produced by the layers in this model is 66.08%. Given the sparsity, we chose to leverage the CSR representation.

❷ **Down-Sampling**: The tensors produced by the max-pooling layers have lower sparsity and higher frequency of unique non-zero values compared to those emitted by ReLU layers. By downsampling the output dimensions, the max-pooling layer further lowers the size of the output tensor. Thus, whenever a maxpooling layer is present in the network, it is more beneficial to compress the output tensor of that layer as opposed to that of the ReLU layer (*i.e.*, before the maxpooling layer).

❸ **Quantization**: Prior efforts have extensively studied how to leverage lower precision arithmetic for inference [11]. We also employ a lower precision representation in our compression scheme. The output tensors of the MultiNN VGG-16 model contain 32-bit floating point numbers. We found that quantizing them to 16-bit floating point numbers did not have a significant impact on accuracy, while cutting the tensor's footprint in half.

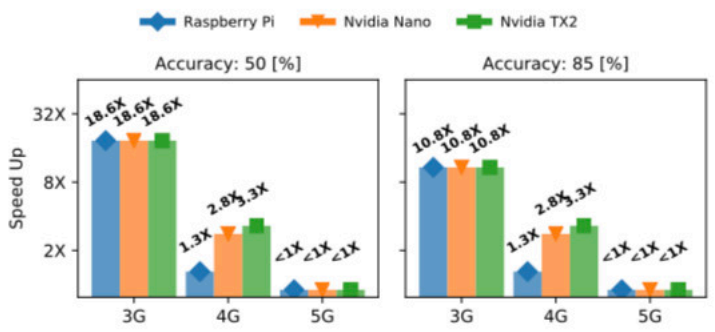
7 EVALUATION

To evaluate our three techniques, we build an end to end system and integrate those techniques with the system. The Section 7.1 introduces the hardware used for our experiments. In Section 7.2, we explain the details of implementing each component in the system. The detail also covers the optimization from the implementation standpoint. Finally, we provide experiments results and insights in Section 7.3.

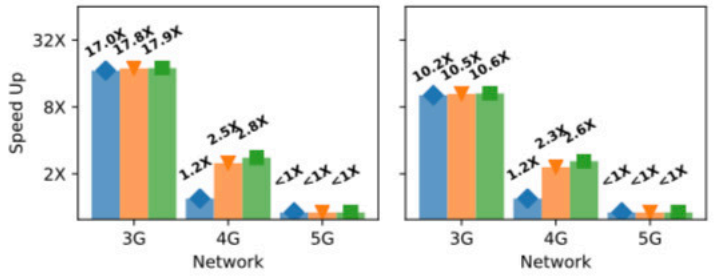
7 Evaluation

We implement our proposed techniques in System X. Our evaluation aims to answer the following questions:

- **MultiNN Model**: Does the MultiNN model allow System X to short-circuit query execution depending on the accuracy requirement? How does it to generalize to multiple DNN models? (§7.3)
- **Lossy Compression:** How effective is the lossy compression scheme compared to lossless compression? How does it affect system throughput? (§7.4)
- **Edge-aware Scheduling:** Can the scheduler determine the optimal plan and execute it for diverse queries, models, and hardware resources? (§7.5)



(a) Throughput Speedup after Compression.



(b) Latency Speedup after Compression.

Figure 11: Compression Impact on Performance.

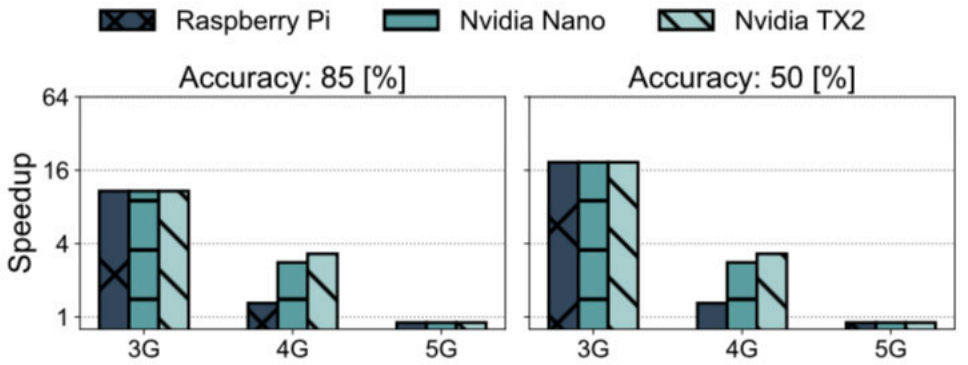


Figure 12: Performance Impact of Compression – The impact of compression schemes on system throughput across different networks and accuracy constraints.

The compression ratio and accuracy tradeoff experiment is conducted on the server offline. We first choose a layer for compression. For example, if layer 3 is chosen, we then use lossy compression technique to compress the output generated after layer 3. We profile the memory usage of the compressed output and compare with the original output to calculate the compression ratio. To get the accuracy, the program then decompresses the lossy compressed output, sends to layer 4 and then continues the DL inference to get the prediction accuracy. For accuracy in compression, we always use the accuracy from the last layer instead early stop points. The reason is because the cloud does not use MultiNN technique as we mentioned in Section 7.3.1. Since the compression technique aims to improve data transfer speed if execution happens on cloud, we do not need to consider early stop accuracy in this case.

Compression Ratio-Accuracy Tradeoff: We first examine the compression ratio across different layers of the MultiNN model. For example, if we pick the third layer, we apply the lossy compression scheme on the output tensor of this layer and measure the compression ratio. We pass the compressed data to the cloud device to finish inference.

We compare four compression schemes: (1) lossless (CSR), (2) lossless (CSR + RLE), (3) lossy (CSR + RLE + count-based dropping), and (4) lossy (CSR + RLE + sequence-based dropping) under different accuracy constraints. The results are shown in Figure 11. The gap between the lossless and lossy compression schemes is more prominent under the lower accuracy constraint setting (50%). For instance, the lossy scheme (CSR + RLE + sequence-based dropping) delivers 18.9× compression ratio on the output tensor of the 12^{th} module as opposed to CSR (3.0×) and CSR + RLE (5.8×) schemes, respectively. In this setting, System X lowers the count and sequence length thresholds to aggressively compress the data. The gap between these schemes shrinks under the higher accuracy constraint setting (85%). In this case, the lossy scheme achieves 8.5× compression ratio as opposed to CSR + RLE (5.8×). All schemes are able to satisfy both accuracy constraints. The output tensors of later layers compress better than those of earlier layers. We attribute this to downsampling by earlier layers of the model that increases the sparsity and dimensionality of these tensors.

SUMMARY

- Leverage tools
  - https://github.com/jarulraj/checker
- Pay attention to visual elements
- Learn from well-written papers