TensorFlow Enabled i2b2 Emulator

Xiao Dong[1] PhD, Eugene M. Sadhu[1] MD, Subhash Kolar Rajanna[1] MD,
David A. Randolph[1] MEng, Edward Barbour[1] MS
[1]Center for Clinical and Translational Science, University of Illinois at Chicago

Introduction

i2b2[1] is an open source software product deployed in many academic medical centers across the country. Typical data dimensions incorporated into i2b2 include clinical and administrative data from electronic medical record systems, diagnosis and procedure codes from billing systems, tissue data from bio-repositories and genetic information. The primary use of i2b2 is cohort identification. Its prevalence and rich data elements made it an appealing target for the purposes of predictive analytics and deep learning.

Deep learning has recently become the paradigm of choice to empower successful artificial intelligence (AI) systems. Two key factors attributed to its groundbreaking success are: first, large enough datasets suitable for neural network analysis finally becoming available; second, hardware platforms that accelerate computation have become widely adopted, such as NVIDIA's Graphical Processing Unit[2] (GPU). In medicine, deep learning's primary contributions to date have been in image analysis, notably in lesion analysis for skin cancer classification[2] and histopathologic slide analysis for prostate cancer diagnosis[3].

In this work we demonstrate the feasibility of replacing i2b2's relational database backend with a deep learning-derived backend. TensorFlow is a popular deep learning framework developed by Google for deep neural network and machine learning projects. TensorFlow uses a computation graph with nodes representing operations and edges, tensors. Many common data models (such as i2b2, OMOP, PCORNet) are EAV based (entity-attribute-value). For a production i2b2 instance, the set of concepts (attributes) may easily number in the hundreds of thousands, typically including ICD, CPT, LOINC, RxNorm and SNOMED codes. Our main intuition to adopt TensorFlow arose from the insight that EAVs such as rows in i2b2's "observation fact" tables may be represented as a 2D sparse tensor, the dense format of which maps all the concepts onto the row axis and the encounters onto the column axis. The SQL operations are then expressed as operations such as matrix multiplication and element-wise array operations. The result is a system which allows the regular cohort identification queries to be seamlessly emulated using TensorFlow, while also opening up the environment to benefit from the built-in GPU acceleration and the rich set of predictive analytics and deep learning tools in the TensorFlow ecosystem.

Methods

The first step is to transform the i2b2 data into a tensor format. The main data content of the i2b2 data mart resides inside one or more "observation fact" tables. To construct the aforementioned 2-dimensional sparse tensor, we first enumerate the encounters, encoding each as an integer. Similarly we enumerate the universe of concept codes, thus mapping each row of observations to an encounter-concept index pair [x,y]. The value associated with the observation fact is encoded in the following manner: for boolean, nominal concepts such as diagnosis and procedure codes, we set the value to 1; for numerical values such as lab tests or vitals, the value is conserved as is; finally, for categorical or ordinal values represented as text such as urine color, when the number of categories is <64, we map each to a bit position, and if ≥64, each is mapped onto consecutive integers beginning with 1. (We note that we could have chosen to further enrich the data by incorporating the modifier codes, such as primary or secondary diagnosis via construction of additional compound concepts.) Such indices [x, y] along with the encoded observation values are fed into the sparse tensor constructor. We emphasize the importance of using a sparse tensor due to two reasons: first, the excellent computational speed achieved through TensorFlow's built-in special operations on them; second, the underlying layout of observation fact data is basically a sparse tensor data structure thus making the data conversion effort trivial. The following simple illustration demonstrates the utility of tensor operations.

The 2-dimensional tensor is a simplified dense tensor representation of 5 encounters and 4 concepts after performing the above transformation. Suppose the concepts consist of ICD-9 codes with two asthma ICD-9 codes (e.g. 493.20, 493.90) corresponding to the 1st and 3rd elements. A query for those two asthma

$$\begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

codes is represented as the 1-dimensional tensor with 1st and 3rd element set to 1, and the product represents the number of matching criteria by encounter. The next step is to emulate regular query executions by composing computation graphs that perform tensor operations. To illustrate, we use the following example: "find all the asthma patients who had a visit between 2016 and 2017 and whose hemoglobin and BMI are both within the normal range". To compute the solution, four consecutive steps are composed: diagnosis attrition, lab and vital value attrition, and visit time attrition.

Numerical comparisons needed for lab and vital attritions were achieved using greater and less than operations. Since TensorFlow does not natively support time comparison, the admit time was encoded as a 64 bit integer. All four attrition operations yield one-dimension tensors that have the exact same size – the total number of encounters. Element-wise array operations were used to generate the final result. Since TensorFlow also supports Logical OR and NOT, all query constructs in i2b2 may be emulated. Figure 1 illustrates the computation graph for the example.

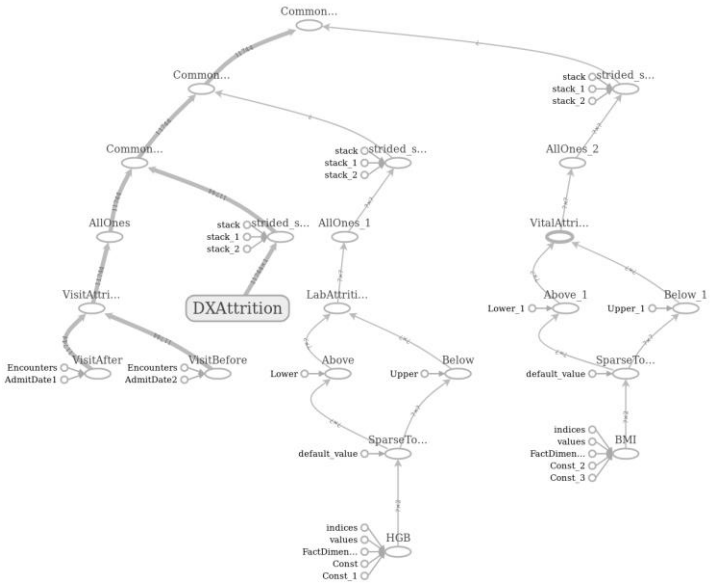

Figure 1. Computational graph to emulate the example query, generated using TensorBoard

Results

Using the methods explained in this paper, we found a computation graph may be composed using tensorflow operators to emulate any i2b2 query pattern, indicating a TensorFlow computation graph composer may be integrated directly into the i2b2 as a drop-in replacement for the relational query engine while keeping front-end and middle-ware layers intact. We have implemented TensorFlow graphs like the one illustrated in Figure 1 on a mid-grade system with a NVIDIA GTX 6GB GPU. As a result, we have a dual-purpose system that is not only capable of supporting regular queries, but also deep learning enabled. We are currently developing meaningful use cases such as generating high quality synthetic data using encoder-decoder RNN.

Discussion

Although our approach is presented in the context of i2b2, it generalizes and is applicable to many other datamarts such as OMOP and PCORNet. The key insight is that since these datamarts all employ the common underlying data structure of [encounter, concept, value], their data contents can be converted directly into sparse tensor format. The future direction for this work is twofold: 1) develop a CRC cell plugin to automatically compose TensorFlow computation graphs, allowing the i2b2 application to run directly on a deep learning enabled backend; 2) utilize the rich set of technical assets in the TensorFlow eco-system to enhance i2b2 with cutting-edge predictive analytics and deep learning capabilities.

References

1. Murphy S, Mendis M, Berkowitz D, *et al*. Integration of clinical and genetic data in the i2b2 architecture. *AMIA Annu Symp Proc.* 2006:1040.
2. Esteva A, Kuprel B, Novoa R, *et al.* Dermatologist-level classification of skin cancer with deep neural networks. *Nature*. 2017;542:115-8.
3. Litjens G, Sánchez C, Timofeeva N, *et al*. Deep learning as a tool for increased accuracy and efficiency of histopathological diagnosis. *Scientific Reports*. 2016;6.