Zehra Sura ([zsura@us.ibm.com](mailto:zsura@us.ibm.com))

Tong Chen ([chentong@us.ibm.com](mailto:chentong@us.ibm.com))

Hyojin Sung (hsung@us.ibm.com)

1. Invention title

Modular Construction of Machine Learning Models for Structured Inputs

1. Invention description - Background
2. Problem statement

Designing machine learning models, particularly neural networks for deep learning, is a trial-and-error process, and more akin to practitioner’s art than science. For application domains that use highly structured input data, we want to use and incorporate the structure of the input data as an integral part of the model design.

1. Prior art

Typically the working of the machine learning model is a black box. In the case of deep neural network models, there has been some work to introspect the network after training to correlate high-level operational semantics to individual components of the network (e.g. in face recognition, and NLP). There has been work to structure networks such that individual layers have access to different permutations/combinations of the input data (e.g. ResNets and DenseNets). There has also been work on Attention Networks that allow some layers of the neural network structure to focus on a part of the input data.

1. Limitations of prior art

None of the existing techniques use a machine learning model structure that is based on and adapts to the structure of the input data.

1. Invention description – Summary
2. Main idea

We describe a method to construct a machine learning model where all input data items to the model are defined by some grammar. The grammar is a collection of entities (individual tokens and expressions, or subsets of tokens/expressions). There is one model for each grammar entity, responsible for learning the function corresponding to that entity, which we refer to as GE-FN (grammar entity function). An input data item is some number of grammar entities arranged in some format (sequence, stack, tree, graph). As the input data is traversed, the machine learning model structure is dynamically traversed using GE-FNs as components that are interconnected by rules that are specific to the input data format.

1. Advantages

Structure and context from input data are intrinsically carried over into the structure of the machine learning model, and do not have to be learned. The constructed machine learning model modular; it is an aggregation of smaller models, each of which can be of a different type (for example, they can be neural networks or decision trees), and each one can be trained more efficiently. The machine learning models constructed using this technique are more amenable to transfer learning and curriculum learning. They are also more amenable to tuning/inspection of the machine learning model to isolate components that do not work as intended or are not accurate enough. Further, properties associated with different grammar entities do not need a uniform encoding (since different sub-models are used), and this can make the overall data representation and learning more efficient.

1. Invention description – Details

The method described applies to learning problems where the input data is structured, i.e. the input can be defined by some grammar. An example application is where the input data are computer programs in some programming language.

The method implicitly carries over the structure of the input data into the structure of the machine learning model that is used for learning. It is particularly useful in cases where there is not enough data to train a single monolithic model, and where the structure of data is both complex and meaningful. For example, computer programs have multiple nested loops where the loop structure can contribute information pertinent to the learning problem.

We address the problem of designing a machine learning model to learn some function F(x), where x belongs to a grammatically structured input domain.

The method works by first applying domain knowledge to find grammar entities that are relevant to the learning problem. For example, consider the problem of estimating the dynamic instruction count of a computer program with basic blocks and loops. The relevant grammar entities can be basic blocks (BB), loop start token (LSTART), and loop end token (LEND).

Second, the input data is formatted in some arrangement of grammar entities, with each entity possibly annotated with some extra property information. In our example problem, the arrangement is a simple sequence, with BB grammar entities annotated with an instruction count, and LSTART and LEND entities annotated with a loop iteration count. The counts in the input data sequence can be obtained using a compiler or dynamic profiler. For example, an input data item can be “BB 10 LSTART 20 BB 5 LEND 20 BB 15”.

Third, each grammar entity is mapped to a function to be learned (and a corresponding machine learning model that learns the function, which we call GE-FN, i.e. grammar entity function). Each function takes two inputs: the current state vector and an annotated property input value. Each function produces one output: the next state vector. For example, BB is mapped to function f1, LSTART to function f2, and LEND to function f3. The machine learning models corresponding to f1, f2, and f3 are smaller models that the final machine learning model will be composed of. The number and size of these component machine learning models can be customized using domain knowledge and appropriately selecting grammar entities in the first step. It is possible to have flexible mappings from grammar entities to functions, e.g. one-to-one or many-to-one mappings.

Fourth, there are rules associated with the input data format that define how to compose the functions (machine learning models) associated with each of the grammar entities in an input data item. In our example where the format is a simple sequence, the output state vector of the preceding function is connected to the input state vector of the succeeding function in the sequence of functions corresponding to the grammar entities. Thus, the output value for the example input data x = “BB 10 LSTART 20 BB 5 LEND 20 BB 15” is computed as F(x) = f1(f3(f1(f2(f1(initial, 10), 20), 5), 20), 15), where “initial” is a preset initial value for the state vector.

The machine learning model resulting from the composition of individual GE-FNs is the final machine learning model used for the learning problem. Note that the model reuses the individual GE-FNs across all inputs, but the operation of the overall machine learning model is dependent on and customized to each instance of the input data. In the case when neural networks are used to model some GE-FNs and backward propagation is used for training the machine learning model, each GE-FN has to have an inverse relationship defined for it (either the corresponding function is differentiable, or it is a mathematical function with a valid inverse function across the range of the input domain, or an inverse relationship is explicitly defined as part of the model design). Then the overall machine learning model, that computes function F(x) and is composed of some component models that are neural networks, can be trained using backward propagation.

Note that domain knowledge can be applied to determine the format of the input data and the associated rules that determine how individual GE-FNs are composed together. For example, the input data can be arranged in a stack, tree, or graph format instead of a simple sequence. In our example, we can customize nested LSTART and LEND grammar entities to follow stack rules that push or pop the value of the state vector that is the output or input of the corresponding GE-FN.

Our method of modularly constructing the structure of the machine learning model has the following advantages:

* Structure and context from input data are intrinsically carried over into the structure of the machine learning model, and do not have to be learned. This can particularly help learning in cases where the input data has long sequences or deep nesting levels.
* The constructed machine learning model is an aggregation of smaller models, each of which can be trained more efficiently.
* The constructed machine learning model is more amenable to curriculum training and transfer learning.
* The constructed machine learning model helps break down the complexity and is more amenable to tuning/inspection of the model to isolate components that do not work as intended or are not accurate enough.
* Properties associated with different grammar entities do not need a uniform encoding (since different sub-models are used), and this can make the overall data representation and learning more efficient.

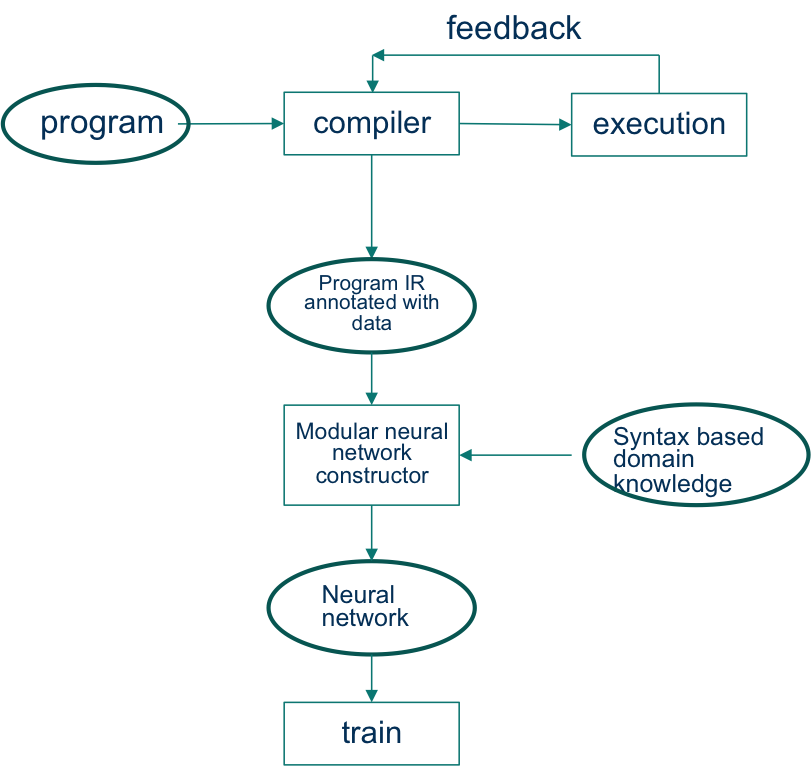


Figure : System Diagram for Example Use Case of Computer Programs as Input

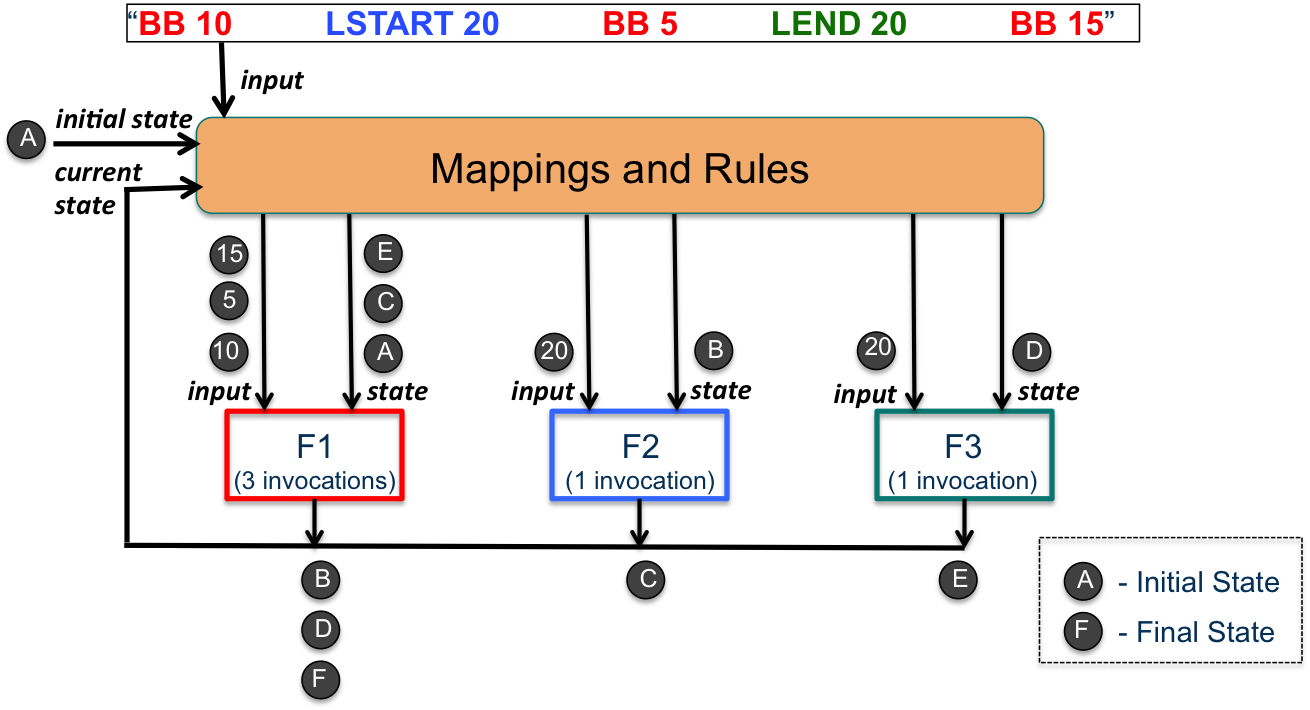


Figure : Structure of Modular Machine Learning Model for an Example Input Data Instance

1. Novelty
2. We introduce a modular machine learning model whose structure is dependent on the structure of the input data. We define a method to construct such a model for grammar-based input data. This model implicitly captures the structure of the input data.
3. Our modular machine learning model is composed of smaller models called GE-FNs, each of which is associated with a grammar entity (grammar token, expression, or subset of tokens/expressions). Each GE-FN learns a function that takes the current state vector as input (with possibly other input data), and produces the next state vector as output.
4. The number and size of GE-FNs can be varied according to problem requirements and domain knowledge, and can be customized by appropriately choosing grammar entities.
5. GE-FNs can be functions that are known apriori, or functions that are to be learned.
6. The composition of the overall machine learning model follows rules that are based on the format of the input data (sequence, stack, tree, etc), and can be used to capture domain knowledge or semantics specific to the learning problem.
7. The GE-FNs can be individually trained using a targeted training input data set, and can be easily adapted to techniques that use curriculum training or transfer learning.
8. The overall machine learning model structure is traversed specific to each input data item, but the components used in the machine learning model are trained across the input set.
9. Related Work:
10. **Parsing Natural Scenes and Natural Language with Recursive Neural Networks**

Use of RNNs to discover the recursive structure in input data

<http://www.socher.org/index.php/Main/ParsingNaturalScenesAndNaturalLanguageWithRecursiveNeuralNetworks>

1. **Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning**

Recent work on ResNets, trying to optimize structure of the network statically

<http://www.aaai.org/ocs/index.php/AAAI/AAAI17/paper/download/14806/14311>

1. **Google’s Neural Machine Translation System: Bridging the Gap Between Human and Machine Translation**

<https://research.google.com/pubs/pub45610.html>

1. **Attention Is All You Need**

Attention networks structure the neural network to focus on certain parts of the input data

<https://arxiv.org/abs/1706.03762>

1. **Curriculum learning**

<https://dl.acm.org/citation.cfm?id=1553380>

1. **Transfer learning**

*Pratt, L. Y. (1993).* [*"Discriminability-based transfer between neural networks”*](http://papers.nips.cc/paper/641-discriminability-based-transfer-between-neural-networks.pdf) *,* [*NIPS Conference: Advances in Neural Information Processing Systems 5*](https://books.google.com/books?id=6tGHlwEACAAJ&pg=PA204)