Relation of work project for Intelligent Systems course

Compressed Network in EML

Ugo Marchesini 1

|  |
| --- |
| **Citation:** Lastname, F.; Lastname, F.; Lastname, F. Title. *Appl. Sci.* **2022**, *12*, x. https://doi.org/10.3390/xxxxx  Academic Editor: Firstname Lastname  Received: date  Accepted: date  Published: date  **Publisher’s Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.    **Copyright:** © 2022 by the authors. Submitted for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/). |

1 Affiliation 1; ugo.marchesini@studio.unibo.it

**Abstract:** le reti neurali sono delle applicazioni del Machine Learning , permettono di risolvere problemi di regressione e classificazione modificando i propri pesi e bias durante la fase di addestramento. Reti più compresse in generale premettono di risolvere problemi con dataset più complessi , ma a costo dell’aumento del footprint della rete in termini di tempo necessario per l’addestramento , la predizione , la dimensione su disco e il carico della rete in caso di applicazione eseguita in remoto. Anche reti neurali con precisione molto elevata non forniscono una chiara comprensione della natura del problema né del modo in cui la rete “risolve” il problema.Una rete neurale modifica i propri pesi e bias per meglio adattare il valore predetto con il valore atteso al fine di migliorare la precisione della predizione, è chiaro che all’aumentare della dimensione dei parametri addestrabili aumenti il costo in tempo ed in energia dell’addestramento stesso. Reti come le CNN (ResNet,Inception,VGG) contengono fino a decine di milioni di pesi , la loro precisione può essere molto alta , ma è anche molto alto il loro footprint. Le tecniche di compressione delle reti neurali permettono di ottenere delle footprint ridotte con perdite accettabili di precisione. Questo non risolve i problemi di ottimizzazione , dove è necessario imporre una funzione obbiettivo e definire dei vincoli tra variabili.EML permette in incorporare un modello ML in un problema combinatorio , in questo caso una rete neurale. Un problema combinatorio essendo dichiarativo potrebbe permettere una migliore comprensione e quindi una migliore “spiegabilità” del problema appreso da una rete neurale. Il campo di applicazione di questa ricerca è la transprecision computing. Lo scopo di questa ricerca è l’esplorazione delle tecniche di compressione delle reti neurali e la sua incorporazione in un modello di ottimizzazione combinatoria al fine di ottenere una pipeline che processi un dataset ottenuto dall’elaborazione di diverse funzioni di calcolo su diversi hardware.

**Keywords:** rete neurale; machine learning; footprint;compression; ottimizzazione combinatoria;

transprecision computing

1. Transprecision computing

Transprecision Computing is a paradigm that allows users to trade the energy associated with computation in exchange for a reduction in the quality of the computation results. Guaranteed numerical precision of each elementary step in a complex computation has been the mainstay of traditional computing systems for many years. La possibilità di ottenere una precisione accettabile al di sotto di un errore predeterminato con un utilizzo minore di bit nelle operazione a FP permette un risparmio energetico apre alla possibilità di ottenere una maggiore potenza di calcolo senza una proporzionale costo energetico.

1. Empirical Model Learning

EML can be considered as a technique that allow to encapsulate the learned model in a number of optimization techniques. Library used codifies each neuron as a variable in the combinatorial problem, while each edge is considered as a constraint on the neurons connected.

1. Compressed Neural Network

Tra le varie tecniche di compressione la ricerca ha preso in considerazione : weight pruning e quantization , dove weight pruning si intende mettere a zero il valore del peso. E’ stata utilizzata il TensorFlow Model Optimization Toolkit (TMOT) per entrambe le tecniche. TMOT permette la conversione ad un formato compresso tflite che però non è utile al fine di utilizzarlo all’interno di un problema combinatorio in quanto la libreria EML accetta modelli in Keras o PyTorch che sono incompatibili con il formato tflite.

1. Dataset

Il dataset è stato preso da <https://zenodo.org/record/6575841#.YtE3kS8QNQJ> , con successive integrazioni.

Il dataset è composto da 9 file divisi per benchmark e hardware.

Ci sono 3 tipi di benchmark : Convolution , Correlation e Saxpy , e corrispondono ad altrettanti algoritmi utilizzati per produrre dei sample.

Ci sono 3 tipo di hardware : g100,pc e vm.

Ogni file ha un nome del tipo <benchmark>\_<hardware>.csv

I sample contengono delle variabili (var\_0 , var\_1 … var\_n) il cui contenuto è il numero di bit usati nel calcolo ,l’errore (error) , il tempo impiegato (time) , l’utilizzo medio della memoria (memory\_mean) e l’utilizzo della memoria di picco (memory\_peak).

1. Used Tools

L’ambiente utilizzato è Visual Studio 2022 , il linguaggio Python per la parte di ML e C# per la data preparation. Le librerie sono : TensorFlow\Keras , Emlib e Util.

1. Process

First Phase : Epidemics , BuildTrainPrintSave , Quantization , Pruning , Compress

Second Phase : BuildTrainPrintSave , Pruning

In una prima fase sono state esplorate le tecniche di compressione,quantizzazione e pruning su un problema epidemiologico (first phase) , e successivamente sul problema di transprecision computing (second phase).

Il problema di Epidemics implementato è la versione con n reti neurali di un pipeline esistente presa da un corso del Prof. Lombardi presente su GitHub ([phd-ml-co-2021-02/notebooks at main · phd-course-ml-co-2021/phd-ml-co-2021-02 · GitHub](https://github.com/phd-course-ml-co-2021/phd-ml-co-2021-02/tree/main/notebooks)). La pipeline dell’esempio originale è la seguente.

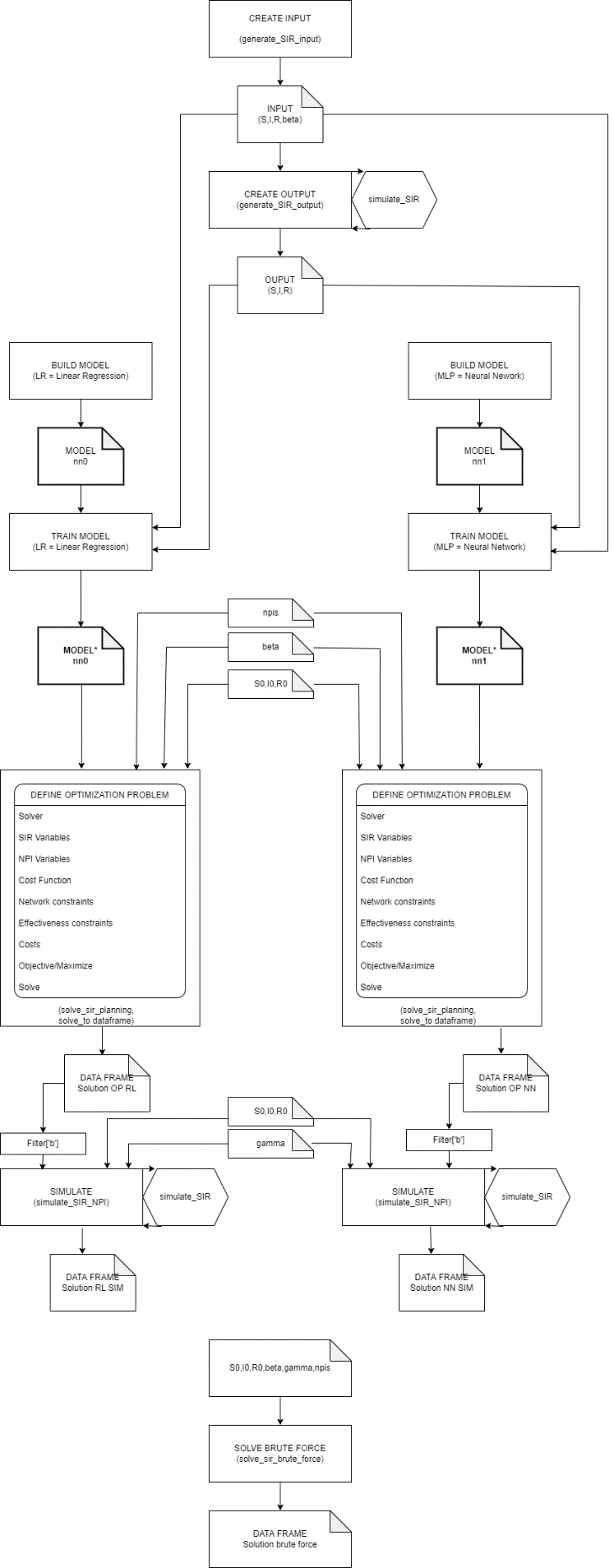


Figura 1 : Original Pipeline Epidemics

La versione con n reti neurali è la seguente.

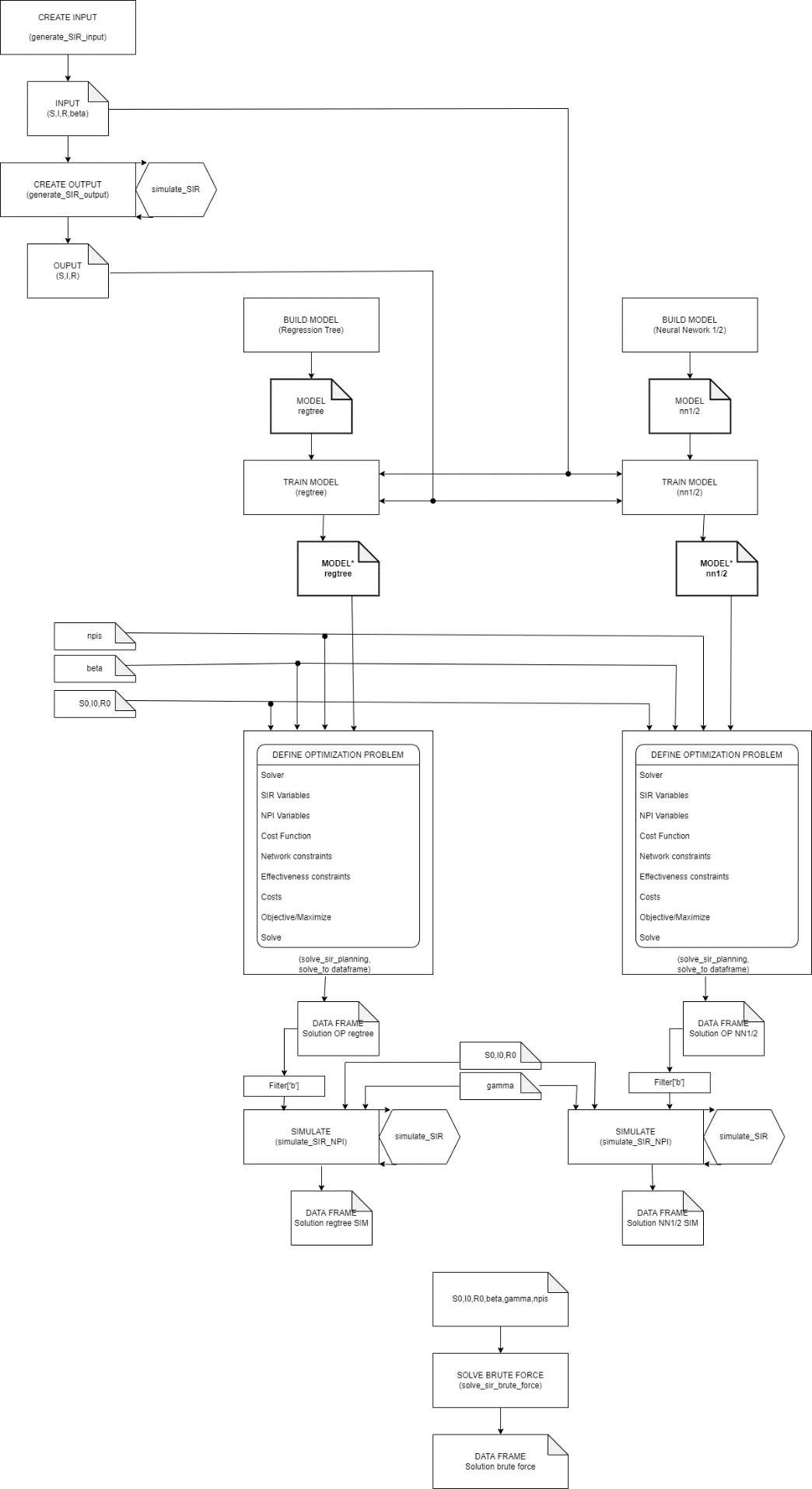


Figura 2 : Pipeline for multiple NN Epidemics

* 1. Epidemics

Il primo esempio su cui esplorare le tecniche di compressione è stato un dataset di un modello epidemico di tipo SIR (Susceptible, Infectious, or Recovered).

Lo scopo del modello è la predizione dei valori SIR e la funzione obbiettivo del problema di ottimizzazione combinatoria è la minimizzazione del costo di un intervento NPI (non pharmaceutil interventions)

* + 1. Build and Train Model

Sono state usate delle funzioni per la creazione dei dati di input e di output configurabili. Le funzioni creano i dati ex novo se richiesto e li salvano , oppure caricano i dati se esistono.

Sono state addestrate due tipi di reti, una con DecisionTreeRegressor e l’altra con modelli Keras con 3 topologie diverse. Sostanzialmente differiscono solo per la profondità e il numero di neuroni per layer.

Tulle le reti sono salvate con estensione h5.

* + 1. Evaluate

La valutazione delle reti è stata effettuata con l’accuracy ottenuta durante la la fase di training , la rmse (root mean square error), mae (mean absolute error) e r2 (coefficient of determination).

Il dataset di test è il 20% del dataset di train.

* + 1. Post Training Quantization

Secondo la documentazione di TensorFlow : “Post-training quantization is a conversion technique that can reduce model size while also improving CPU and hardware accelerator latency, with little degradation in model accuracy. You can quantize an already-trained float TensorFlow model when you convert it to TensorFlow Lite format using the TensorFlow Lite Converter.”

Sono offerte 6 tecniche di quantizzazione : Dynamic range quantization, Full integer quantization, Integer with float fallback, Float16 quantization, Integer only: 16-bit activations with 8-bit weights (V1) e Integer only: 16-bit activations with 8-bit weights + builtins (V2)

Al fine di valutare le metriche di queste tecniche è necessario effettuare il training del modello (baseline) prima della conversione ad un modello quantizzato e compresso in formato tflite.

Le metriche sono : size , rmse, mae e r2;

* + 1. Pruning

TMOT offre la possibilità di effettuare il pruning di una rete addestrata , la dimensione della rete dopo il pruning è maggiore della rete baseline in quanto lo scopo finale è la compressione attraverso la conversione di TFLite.

* + 1. Compress

TMOT permette la compressione di nn in un formato tflite , questo formato non è utile al fine di un inserimento di un problema di ottimizzazione combinatoria offerto dalle librerie emlib in quanto la rete non è un tipo Keras che invece usa emlib.

* + 1. Simulation and Brute Force

Il problema di ottimizzazione combinatoria ha in uscita il vettore beta (infected per day) che è utilizzato insieme ai valori iniziali di S0,I0 e R0 nella funzione di simulazione fornita da emlib.

Altrettanto viene effettuato con la funzione di Brute\_Force per verificare la validità della pipeline di processo del dataset.

* + 1. Conclusion

La metriche rilevate riguardano le dimensioni dei modelli compressi rispetto alla baseline e la precisione delle predizioni. Nella tabella seguente è stata usata una rete con 4 layer e 8 neuroni per layer (4X8).

Si è rilevato che con l’aumento della complessità della rete non è garantito di ottenere una soluzione ottima o fattibile in uscita dal problema combinatorio, per questo è stato scelto un modello relativamente semplice (4X8)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | TrainingTime | size | accuracy | loss | rmse | mae | r2 |
| Reference | 74573 | 22176 | 0.988 | 6.68519e-05 | 0.00817 | 0.00578 | 0.99711 |
| Pruned | 24947 | 33984 | 0.989 | 5.25832e-05 | 0.00725 | 0.00499 | 0.99770 |
| DynamicRangeQuantization |  | 3576 |  |  | 0.00817 | 0.00578 | 0.99711 |
| FullIntegerQuantizationIntegerOnly |  | 3392 |  |  | 2.14685 | 1.62933 | 0.99687 |
| FullIntegerQuantizationIntegerWithFloatFallback |  | 3736 |  |  | 0.00811 | 0.00607 | 0.99708 |
| Float16Quantization |  | 4304 |  |  | 0.00818 | 0.00580 | 0.99710 |
| IntegerOnly16BitActivationsWith8BitWeightsV1 |  | 3576 |  |  | 0.00817 | 0.00578 | 0.99711 |
| IntegerOnly16BitActivationsWith8BitWeightsV2 |  | 3576 |  |  | 0.00817 | 0.00578 | 0.99711 |

La valutazione che è possibile dare è che la compressione offerta da TMOT è interessante ,nonostante non possa essere inserita all’interno di un problema di ottimizzazione a causa della differente struttura dati incompatibile con emlib, e la precisione di una rete pruned non discosta dalla baseline ma ha una dimensione superiore. Nella prossima fase sarà necessario approfondire se diversi tipi di topologie di nn permettono una miglioramento del tempo di solving del problema di ottimizzazione.

Verificata la fattibilità di implementare la stessa pipeline su un ambiente differente ,la pipeline originale è su Jupiter Notebook mentre quella attuale e il problema di Transprecision Computing sono sullo stack VisualStudio\Python, ed acquisito confidenza con le librerie si è passati alla seconda fase.

* 1. Transprecision Computing

L’esecuzione della pipeline viene effettuata con una batteria di reti su tutto il dataset , quindi per ogni rete viene processato la versione baseline e la versione pruned per ogni benchmarck , quindi ci sono 6 log diversi per ogni rete processata.

Le esecuzioni sono definite come “run” , viene creata una cartella con nome run <dataora> che è la destinazione dei log , dei modelli e del run.summary.csv.

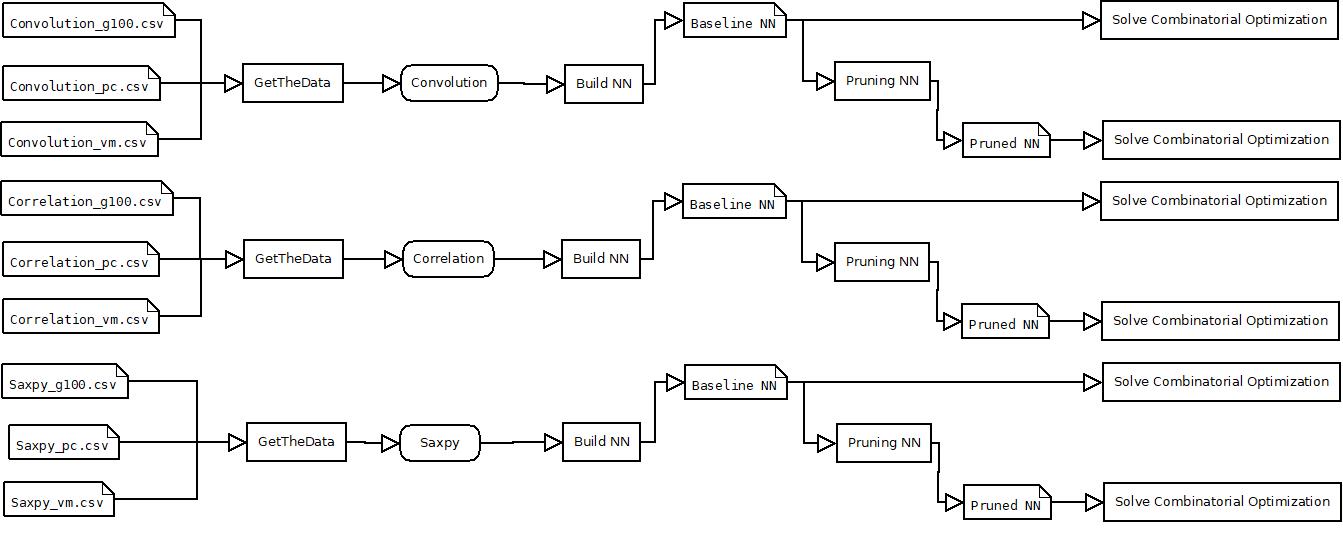


Figura 3: Pipeline Tpc

* + 1. GetTheData

La baseline è divisa in 9 file : 3 per benchmark (Convolution,Correlation e Saxpy) e 3 per hardware (g100,pc e vm).

I modelli sono orientati al benchmark quindi è stata sviluppata una funzione che trasforma il dataset di ogni benchmark in un dataframe one-hot-encoding per i valori di g100,pc e vm.

I valori sono stati normalizzati nell’intervallo tra 0 e 1 , è stata tenuta memoria dei valori di upper e lower bound per denomalizzare i risultati predetti.

* + 1. Build and Train Model

Come nella fase precedente è stato necessario l’addestramento di varie reti utilizzando la funzione present in Util. In questa fase è ottenuta l’accuracy , loss , traning time e size.

* + 1. Evaluate

Le metriche ottenute sono : evaluation time, rmse, mae e r2

* + 1. Pruning

Come per Epidemics viene effettuata la pruning della rete. I parametri utilizzati sono initial\_sparsity e final\_sparsity ed indicano la percentuale di pesi da porre a 0 all’inizio e alla fine del processo di pruning.

* + 1. Encoding

La funzione obbiettivo è la minimizzazione del numero di bit delle variabili , con dei vincoli sul time e sull’error.

* + 1. Metrics

Le funzioni di build , pruning e encoding emettono dei log dettagliati per ogni modello e per ogni benchmark e parte di questi valori sono raccolti in una file chiamato run.summary.csv.

La seguente tabella contiene le metriche raccolte nel file run.summary.csv

|  |  |
| --- | --- |
| **Nome** | **Descrizione** |
| **Topology** | Descrive la topologia è del tipo : <neuroni layer><spazio><neuroni layer>…<neuroni layer> |
| **Train** | Tempo in ms dell’addestramento |
| **Size** | Byte del file h5 |
| **Evaluate** | Tempo in ms delle metriche rmse,rme e r2 |
| **Accuracy** | Precisione ottennuta dalla fase di addestramento |
| **Rmse** | Root mean square error |
| **Encode** | Tempo in ms per l’encoding nel promplema di ottimizzazione |
| **Solver** | Tempo in ms di soluzione del problema di ottimizzazione |
| **Objective** | Risultato della funzione obbiettivo (sommatoria delle variabili di input var\_x) normalizzata |
| **Status** | Stato della soluzione del probkema di ottimizzazione (OPTIMAL,FEASIBLE …) |

* + 1. Result

La valutazione viene effettuata su una run specifica (run20220901124749) in cui sono state usate 6 reti diverse.

E’ possibile fornire alcune valutazioni :

* Reti più grandi non forniscono una accuracy migliore
* Il dataset Saxpy ha un’accuracy migliore
* Le reti pruned hanno un tempo di soluzione del problema di ottimizzazione minore della corrispondente baseline

La topology è il numero dei pesi della rete.

I punti rossi sono problemi INFEASIBLE , i punti gialli NOT\_SOLVED , tutti gli altri sono soluzioni ottime.

Nei grafici <benchmark> solver/topology le ascisse sono il log10(topology) e le ordinate il log10(solver).

E’ possibile concludere che le reti pruned permettono sempre ,quando ottengono un risultato OPTIMAL, dei tempi migliori rispetto alle corrispondenti reti baseline.

1. Repository
2. Architecture
3. Roadmap

3.

References

References must be numbered in order of appearance in the text (including citations in tables and legends) and listed individually at the end of the manuscript. We recommend preparing the references with a bibliography software package, such as EndNote, ReferenceManager or Zotero to avoid typing mistakes and duplicated references. Include the digital object identifier (DOI) for all references where available.

Citations and references in the Supplementary Materials are permitted provided that they also appear in the reference list here.

In the text, reference numbers should be placed in square brackets [ ] and placed before the punctuation; for example [1], [1–3] or [1,3]. For embedded citations in the text with pagination, use both parentheses and brackets to indicate the reference number and page numbers; for example [5] (p. 10), or [6] (pp. 101–105).

1. Author 1, A.B.; Author 2, C.D. Title of the article. *Abbreviated Journal Name* **Year**, *Volume*, page range.
2. Author 1, A.; Author 2, B. Title of the chapter. In *Book Title*, 2nd ed.; Editor 1, A., Editor 2, B., Eds.; Publisher: Publisher Location, Country, 2007; Volume 3, pp. 154–196.
3. Author 1, A.; Author 2, B. *Book Title*, 3rd ed.; Publisher: Publisher Location, Country, 2008; pp. 154–196.
4. Author 1, A.B.; Author 2, C. Title of Unpublished Work. *Abbreviated Journal Name* year, *phrase indicating stage of publication (submitted; accepted; in press)*.
5. Author 1, A.B. (University, City, State, Country); Author 2, C. (Institute, City, State, Country). Personal communication, 2012.
6. Author 1, A.B.; Author 2, C.D.; Author 3, E.F. Title of Presentation. In Proceedings of the Name of the Conference, Location of Conference, Country, Date of Conference (Day Month Year).
7. Author 1, A.B. Title of Thesis. Level of Thesis, Degree-Granting University, Location of University, Date of Completion.
8. Title of Site. Available online: URL (accessed on Day Month Year).