# DEEP LEARNING PER LA MATEMATICA SIMBOLICA

**Elia Mercatanti**

Relatore: *Donatella Merlini*

Università degli Studi di Firenze
Scuola di Scienze Matematiche, Fisiche e Naturali
Corso di Laurea in Informatica

Anno Accademico 2020-2021

# Motivazioni

**Successo delle reti neurali negli ultimi anni**

- Sono lo stato dell'arte in un ampia varietà di problemi.
- Sono estremamente efficaci nel *pattern recognition*.
- Limitato successo nel calcolo simbolico, anche in compiti semplici come la moltiplicazione di interi.
- Grande successo su compiti del *Natural Language Processing* e sulle traduzioni: problemi di manipolazione simbolica.

**Applicare il *deep learning* al calcolo simbolico**

- Anche le persone hanno difficoltà nell'eseguire complessi calcoli simbolici.
- Il *pattern recognition* può essere utile per l'integrazione.
- Gli approcci precedenti hanno quasi sempre considerato dataset molto piccoli.

**Architetture per la Traduzione Automatica**

- Lavorano su frasi considerate come sequenze di *tokens*.
- Non hanno bisogno di specifiche informazioni sul problema.
- Usano enormi dataset.

**Matematica Simbolica**

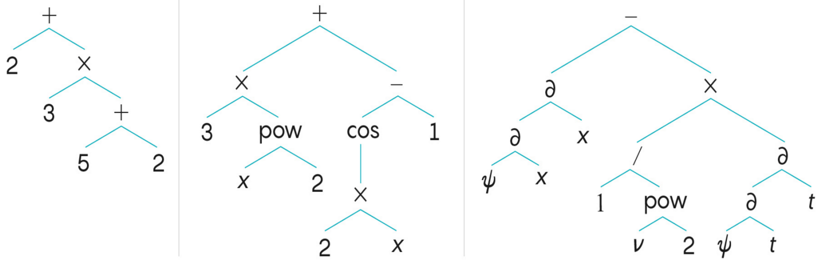
- Può essere considerata come un linguaggio.
- Possono essere generati grandi dataset.
- Risolvere un problema equivale a "tradurre" quest'ultimo nella sua soluzione.

**Cosa è stato fatto**:

- Sono state usate tecniche per il *Natural Language Processing* sui problemi di matematica simbolica.
- Su due problemi: integrazioni, equazioni differenziali.
- Usando modelli *Sequence to Sequence* (Seq2Seq), in particolare il *Transformer*.
- Problemi e soluzioni rappresentati tramite sequenze.
- Generando grandi dataset di problemi e soluzioni.
- Addestrando vari modelli per "tradurre" i problemi nelle loro soluzioni.
- Valutando le loro prestazioni, anche su alcuni casi particolari.
- Confronto con framework di calcolo simbolico classici (*Mathematica*, *Maple* e *Matlab*)

# Espressioni Matematiche in Forma di Alberi

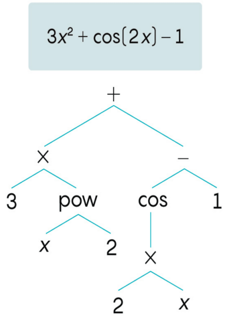
$2 + 3 \times (5 + 2)$ | $3x^2 + \cos(2x) - 1$ | $\frac{\partial^2 \psi}{\partial x^2} - \frac{1}{\nu^2} \frac{\partial^2 \psi}{\partial t^2}$




**Vantaggi**:

- Non ambiguità nell'ordine delle operazioni.
- Rimozione di simboli non significativi (parentesi, spazi, ecc.).
- Ad ogni espressione diversa corrisponde un albero diverso.
- Corrispondenza biunivoca tra espressioni ed alberi.

# Dagli Alberi alle Sequenze




Sequenza in Notazione Prefissa:

$[+ \ \times \ 3 \ \ \text{pow} \ \ x \ \ 2 \ - \ \cos \ \times \ 2 \ \ x \ \ 1]$

**Vantaggi sull'uso della notazione prefissa**:

- Trasforma facilmente gli alberi in sequenze.
- Garantisce rappresentazione biunivoca tra sequenze ed alberi.
- Non necessita di parentesi finché ogni operatore ha un numero fisso di operandi.

# Generare Espressioni Matematiche Casuali

Sono stati generati grandi dataset di espressioni matematiche casuali per i due problemi scelti, sfruttando gli alberi.

- Per ogni tipo di problema viene impiegata una strategia diversa.

Per generare un singolo problema o una soluzione casuale:

- Viene generato un albero unario-binario casuale.
- Per ogni nodo interno vengono selezionati operatori casuali.
- Ogni nodo foglia viene sostituito con una costante, un intero, o una variabile casuale.
- Ogni forma di albero ha la possibilità di essere generata con la stessa probabilità.

# Integrazioni - Forward Generation (FWD)

**Strategia del Generatore**:

- Viene generata un funzione casuale $f$.
- Viene calcolata la sua primitiva $F$ con un framework di matematica simbolica (*SymPy*).
- La coppia $(f, F)$ viene aggiunta al dataset.

**Caratteristiche**:

- Richiede una framework di calcolo simbolico esterno.
- Limitato alle funzioni che il framework può integrare.
- Lento dal punto di vista computazionale.
- Tende a generare problemi corti con soluzioni lunghe.

# Integrazioni - Backward Generation (BWD)

**Strategia del Generatore**:

- Viene generata un funzione casuale $f$.
- Viene calcolata la sua derivata $f'$ con un framework di matematica simbolica (*SymPy*).
- La coppia $(f', f)$ viene aggiunta al dataset.

**Caratteristiche**:

- La differenziazione è sempre possibile ed estremamente veloce.
- Non dipende da un sistema di integrazione simbolica esterno.
- Efficiente dal punto di vista computazionale.
- Tende a generare problemi lunghi con soluzioni corte.
- Improbabile che venga generato l'integrale di funzioni semplici.

# Integrazioni - Backward con Integrazione per Parti (IBP)

**Strategia del Generatore**:

- Vengono generate due funzione casuali $F$ e $G$.
- Vengono calcolate la loro derivata $f$ e $g$ con un framework di matematica simbolica (*SymPy*).
- Se $f * G$ è presente nel dataset, viene calcolato l'integrale di $F * g$ con:

$$\int Fg = FG - \int fG$$

- Il nuovo integrale scoperto viene aggiunto al dataset.

**Caratteristiche**:

- Può generare gli integrali di funzioni semplici.
- Lento dal punto di vista computazionale.