

MergeKit

A Toolkit for Merging Large Language Models

Studente

Alessandro La Cava
Matr. 247436

Indice

- Introduzione
- Dataset
- Model Merging
- MergeKit
- Linear
- TIES
- Breadcrumbs
- SLERP
- NuSLERP
- DARE
- DELLA
- Arcee Fusion
- Analisi dei risultati
- Conclusioni

Introduzione

Obiettivo: studio delle diverse tecniche di model merging e analisi delle performance nella creazione di modelli multitask

Librerie utilizzate:

- unsloth
- mergekit
- lm-evaluation-harness

Modello utilizzato:

- Llama3.1 8B

Dataset

nvidia/OpenMathReasoning

- Circa 5.7 milioni di problemi di matematica, la cui soluzione è formattata secondo lo schema Chain-of-Thought (CoT).

TokenBender/code_instructions_122k_alpaca_style

- Circa 120.000 esercizi di programmazione su diversi linguaggi in formato Alpaca.
- Il Supervised Fine-tuning è stato limitato a un sottounsieme di 10.000 campioni per ciascun dataset.

Model Merging

Il model merging è una tecnica che consente di:

- combinare più modelli specializzati in un unico modello senza la necessità di training aggiuntivo
- trasferire funzionalità tra modelli senza dover accedere ai dati di training
- migliorare le prestazioni mantenendo i costi di inferenza
- trovare compromessi ottimali tra i diversi comportamenti del modello

Esistono diverse approcci:

- Averaging (Linear Interpolation)
- LoRA Merging
- Layer-wise Merging

MergeKit

MergeKit è un tool open source che implementa diversi algoritmi di merging.

Le operazioni da eseguire per effettuare la merge sono specificate come documenti YAML.

Gli elementi principali di un file di configurazione sono:

- merge_method
- slices/models
- base_model
- parameters
- dtype
- tokenizer
- chat_template.

Linear

Media pesata del vettore dei pesi.

```
dtype: float16
merge_method: linear
modules:
    default:
        slices:
            - sources:
                - layer_range: [0, 32]
                    model: Alelcv27/Llama3.1-8B-Code
                    parameters:
                        weight: 1.0
                - layer_range: [0, 32]
                    model: Alelcv27/Llama3.1-8B-Math-CoT
                    parameters:
                        weight: 1.0
```

TIES

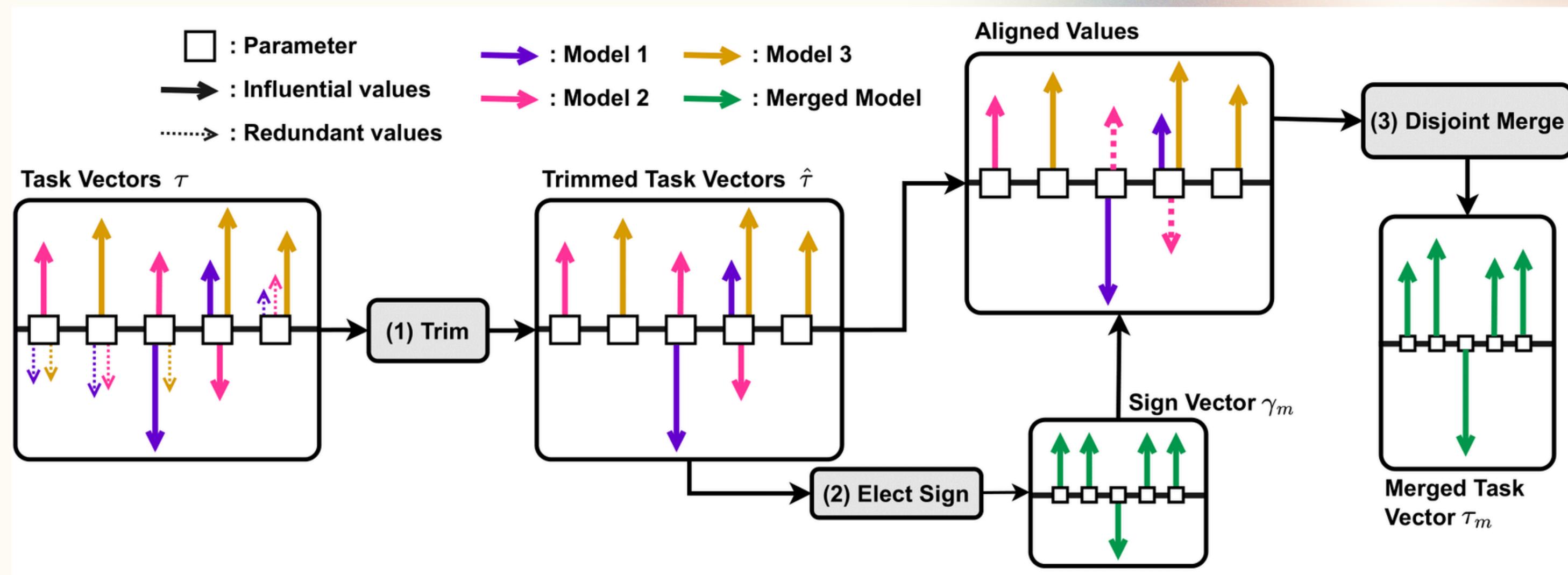
Calcola i task vectors per ogni modello sottraendo i pesi del modello pre-trained dai pesi dello stesso modello ma fine-tuned.

Identifica e rimuove i parametri ridondanti nei singoli modelli.

Utilizza un vettore di segno per risolvere conflitti tra direzioni di cambiamento dei parametri opposte.

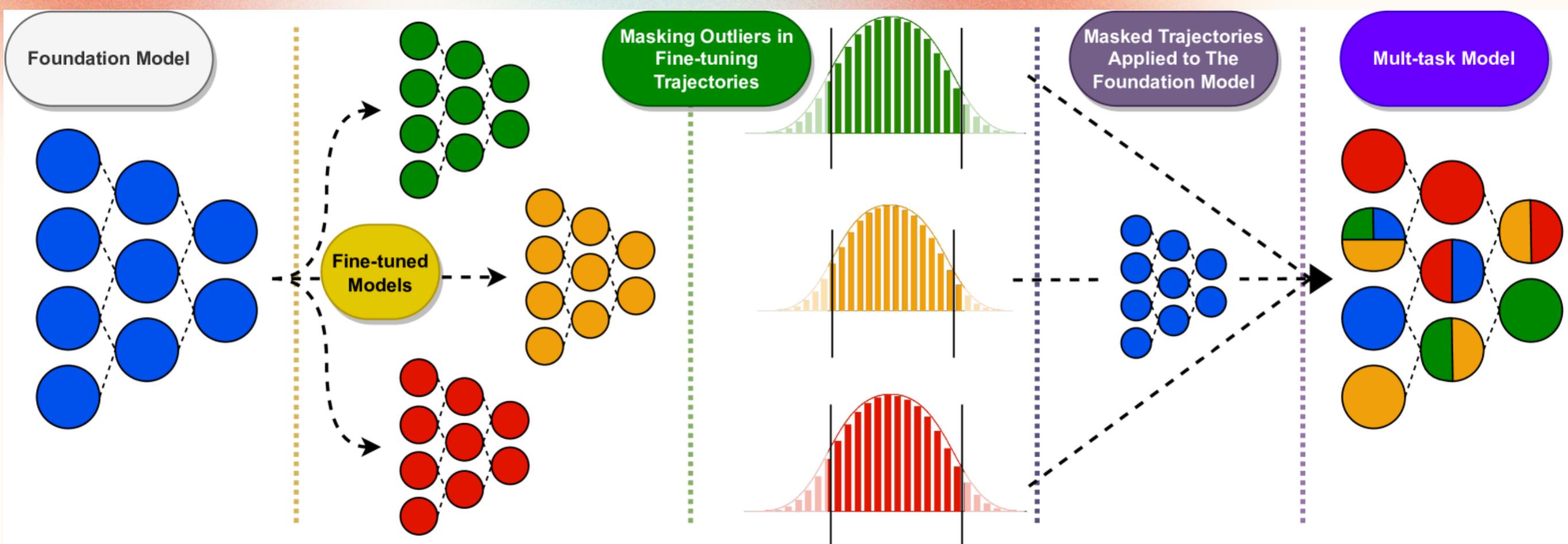
```
base_model: meta-llama/Llama-3.1-8B-Instruct
dtype: float16
merge_method: ties
modules:
  default:
    slices:
      - sources:
          - layer_range: [0, 32]
            model: Alelcv27/Llama3.1-8B-Math-CoT
            parameters:
              density: 0.5
              weight: 0.8
          - layer_range: [0, 32]
            model: Alelcv27/Llama3.1-8B-Code
            parameters:
              density: 0.5
              weight: 0.7
        - layer_range: [0, 32]
          model: meta-llama/Llama-3.1-8B-Instruct
          parameters:
            normalize: 1.0
```

TIES



Breadcrumbs

Estensione dei metodi task arithmetic che scarta sia le differenze estremamente piccole che quelle estremamente grandi rispetto al modello base.



Breadcrumbs

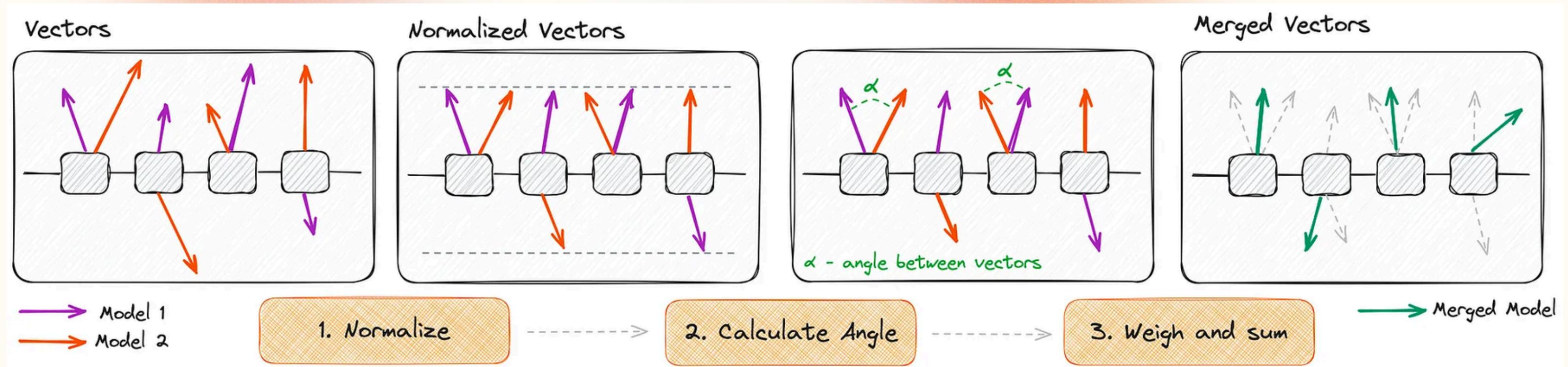
Può essere utilizzato sia con l'algoritmo di consenso del segno (breadcrumbs_ties) che senza (breadcrumbs).

```
base_model: meta-llama/Llama-3.1-8B-Instruct
dtype: float16
merge_method: breadcrumbs
modules:
  default:
    slices:
      - sources:
          - layer_range: [0, 32]
            model: Alelcv27/Llama3.1-8B-Code
            parameters:
              weight: 0.8
      - layer_range: [0, 32]
            model: Alelcv27/Llama3.1-8B-Math-CoT
            parameters:
              weight: 0.8
      - layer_range: [0, 32]
            model: meta-llama/Llama-3.1-8B-Instruct
            parameters:
              density: 0.9
              gamma: 0.01
              normalize: 1.0
```

```
base_model: meta-llama/Llama-3.1-8B-Instruct
dtype: float16
merge_method: breadcrumbs_ties
modules:
  default:
    slices:
      - sources:
          - layer_range: [0, 32]
            model: Alelcv27/Llama3.1-8B-Math-CoT
            parameters:
              density: 0.5
              weight: 0.8
      - layer_range: [0, 32]
            model: Alelcv27/Llama3.1-8B-Code
            parameters:
              density: 0.6
              weight: 0.9
      - layer_range: [0, 32]
            model: meta-llama/Llama-3.1-8B-Instruct
            parameters:
              normalize: 1.0
```

SLERP

Interpolazione sfericamente i parametri di due modelli, uno dei quali deve essere scelto come base_model.



SLERP

Viene implementato utilizzando i seguenti passaggi:

- Normalizza i vettori di input.
- Calcola l'angolo tra i vettori.
- Se i vettori sono quasi collineari, viene utilizzata l'interpolazione lineare. Altrimenti calcola i fattori di scala in base al fattore di interpolazione t e all'angolo tra i vettori.
- Tali fattori vengono utilizzati per pesare i vettori originali, che vengono poi sommati per ottenere il vettore di interpolazione.

```
base_model: Alelcv27/Llama3.1-8B-Math-CoT
dtype: float16
merge_method: slerp
modules:
  default:
    slices:
      - sources:
          - layer_range: [0, 32]
            model: Alelcv27/Llama3.1-8B-Code
          - layer_range: [0, 32]
            model: Alelcv27/Llama3.1-8B-Math-CoT
parameters:
  t: 0.3
```

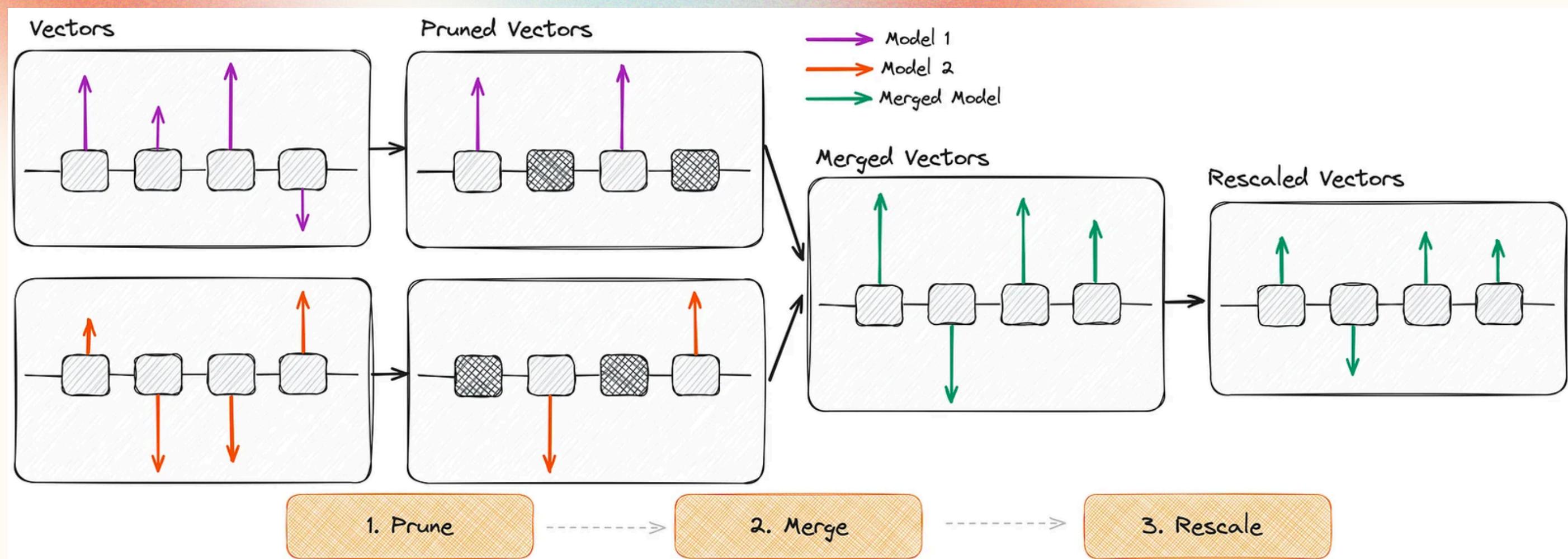
NuSLERP

Estensione di SLERP che esegue l'interpolazione sferica diretta tra due modelli o, in presenza di un base_model, opera sui task vector calcolati per differenza. In questo secondo caso, l'algoritmo applica lo SLERP ai delta dei modelli principali rispetto alla base, sommandoli infine a quest'ultima.

```
base_model: meta-llama/Llama-3.1-8B-Instruct
dtype: float16
merge_method: nuslerp
modules:
  default:
    slices:
      - sources:
          - layer_range: [0, 32]
            model: Alelcv27/Llama3.1-8B-Math-CoT
            parameters:
              weight: 0.6
          - layer_range: [0, 32]
            model: Alelcv27/Llama3.1-8B-Code
            parameters:
              weight: 0.5
      - layer_range: [0, 32]
        model: meta-llama/Llama-3.1-8B-Instruct
```

DARE

Stesso funzionamento di TIES. In più, DARE fa random pruning con rescaling per rispecchiare meglio le performance dei modelli originali.



DARE

DARE può essere usato sia con l'algoritmo di consenso del segno (dare_ties) che senza (dare_linear).

```
base_model: meta-llama/Llama-3.1-8B-Instruct
dtype: float16
merge_method: dare_ties
modules:
  default:
    slices:
      - sources:
          - layer_range: [0, 32]
            model: Alelcv27/Llama3.1-8B-Math-CoT
            parameters:
              density: 0.8
              weight: 0.7
          - layer_range: [0, 32]
            model: Alelcv27/Llama3.1-8B-Code
            parameters:
              density: 0.6
              weight: 0.7
      - layer_range: [0, 32]
        model: meta-llama/Llama-3.1-8B-Instruct
        parameters:
          normalize: 1.0
```

```
base_model: meta-llama/Llama-3.1-8B-Instruct
dtype: float16
merge_method: dare_linear
modules:
  default:
    slices:
      - sources:
          - layer_range: [0, 32]
            model: Alelcv27/Llama3.1-8B-Code
            parameters:
              weight: 0.5
          - layer_range: [0, 32]
            model: Alelcv27/Llama3.1-8B-Math-CoT
            parameters:
              weight: 0.5
      - layer_range: [0, 32]
        model: meta-llama/Llama-3.1-8B-Instruct
```

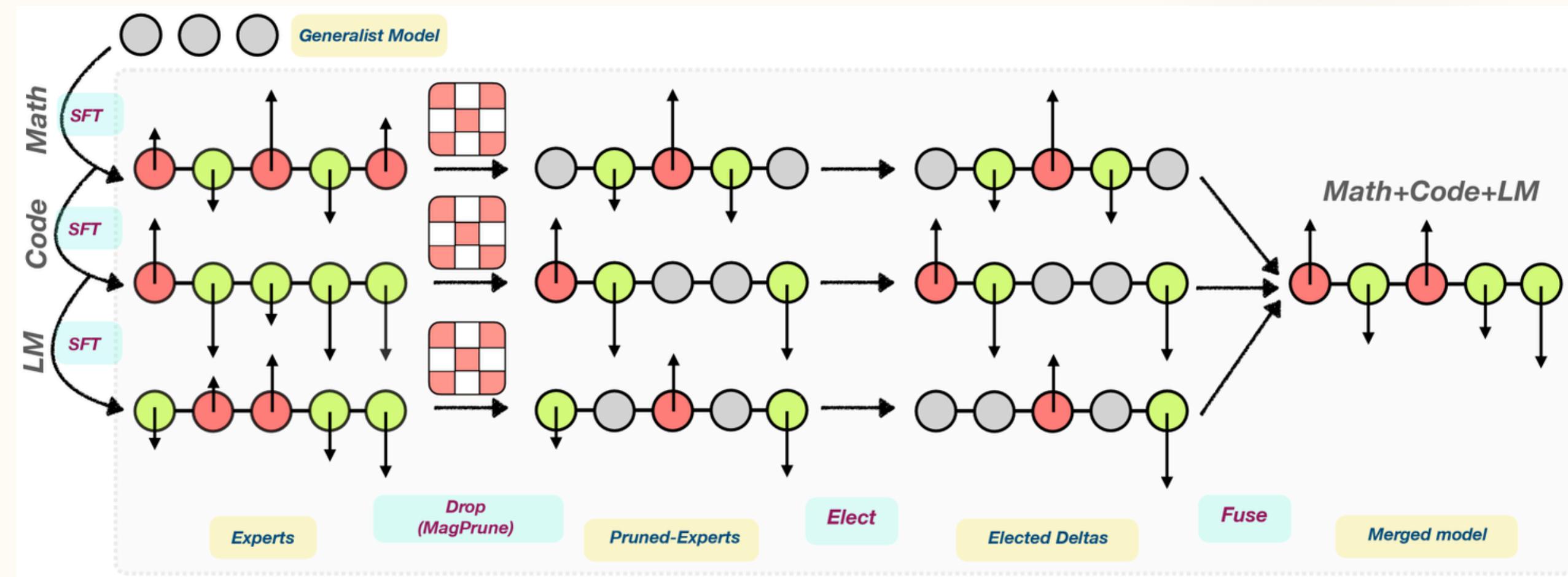
DELLA

Basandosi su DARE, DELLA utilizza un pruning adattivo basato sulla grandezza dei parametri. Classifica, innanzitutto, i parametri di ciascuna riga di parametri delta (differenza tra i parametri fine-tuned e quelli pre-trained) e assegna drop probabilities inversamente proporzionali alla loro grandezza. Ciò gli consente di conservare le modifiche più importanti riducendo le interferenze. Dopo il pruning, ridimensiona i restanti parametri in modo simile a DARE.

```
base_model: meta-llama/Llama-3.1-8B-Instruct
dtype: float16
merge_method: della
modules:
  default:
    slices:
      - sources:
          - layer_range: [0, 32]
            model: Alelcv27/Llama3.1-8B-Math-CoT
            parameters:
              density: 0.8
              weight: 0.7
      - layer_range: [0, 32]
            model: Alelcv27/Llama3.1-8B-Code
            parameters:
              density: 0.7
              weight: 0.5
      - layer_range: [0, 32]
            model: meta-llama/Llama-3.1-8B-Instruct
            parameters:
              epsilon: 0.1
              lambda: 1.0
              normalize: 1.0
```

DELLA

Può essere usato con (della) o senza (della_linear) l'algoritmo di consenso del segno.



Arcee Fusion

Filtra le fusioni di modelli applicando un'integrazione selettiva in tre fasi::

- Determina uno score di importanza per ogni parametro combinando la differenza assoluta tra i modelli con la divergenza di KL.
- Identifica i parametri rilevanti tramite rilevamento di outlier (metodo IQR), filtrando le variazioni meno significative.
- Applica una maschera di fusione per integrare nel modello base esclusivamente i parametri che hanno superato la soglia di importanza.

```
base_model: Alelcv27/Llama3.1-8B-Math-CoT
dtype: bfloat16
merge_method: arcee_fusion
modules:
  default:
    slices:
      - sources:
          - layer_range: [0, 32]
            model: Alelcv27/Llama3.1-8B-Math-CoT
          - layer_range: [0, 32]
            model: Alelcv27/Llama3.1-8B-Code
```

Analisi dei risultati

	humaneval	tinyARC	tinyGSM8K: flexible match	tinyGSM8K: exact match	tiny Hellaswag	tinyMMLU	tiny TruthfulQA	tiny Winogrande
Instruct	0,5793	0,6533	0,7545	0,7304	0,8054	0,6289	0,5392	0,7598
Solo Code	0,5122	0,6458	0,75	0,75	0,819	0,6493	0,5228	0,7441
Solo Math	0,5793	0,6247	0,728	0,728	0,8237	0,6808	0,5567	0,744
DataMerged	0,5122	0,6507	0,7601	0,7601	0,8304	0,6603	0,5383	0,7432
Math → Code	0,4634	0,636	0,7169	0,7098	0,789	0,6669	0,5219	0,7534
Code → Math	0,5427	0,6283	0,7153	0,6537	0,8249	0,6696	0,5439	0,7558
Linear	0,5793	0,6536	0,778	0,778	0,8235	0,6646	0,5514	0,7503

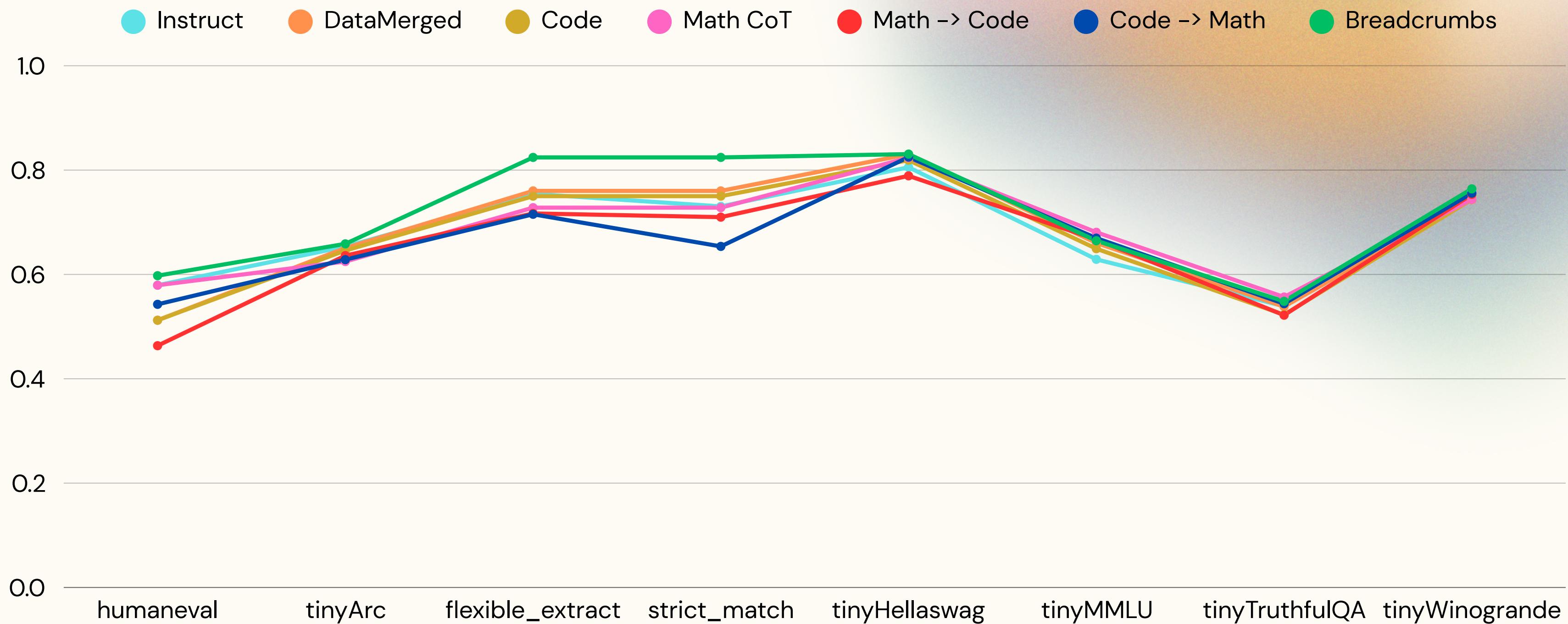
Analisi dei risultati

	humaneval	tinyARC	tinyGSM8K: flexible match	tinyGSM8K: exact match	tiny Hellaswag	tinyMMLU	tiny TruthfulQA	tiny Winogrande
TIES	0,5488	0,6291	0,727	0,727	0,8191	0,6679	0,5464	0,7368
Dare Linear	0,5793	0,6536	0,778	0,778	0,8235	0,6646	0,5512	0,7503
Dare Ties	0,5244	0,6302	0,7627	0,7627	0,8191	0,6496	0,545	0,7478
Arcee Fusion v1	0,5427	0,6586	0,7659	0,7659	0,8237	0,6668	0,5567	0,7415
Arcee Fusion v2	0,5549	0,6378	0,7896	0,7896	0,8297	0,6648	0,5467	0,7656
DELLA	0,5793	0,6291	0,7174	0,7174	0,8202	0,6849	0,5542	0,7368
NuSLERP	0,561	0,6291	0,7334	0,7334	0,8238	0,6812	0,5503	0,7528

Analisi dei risultati

	humaneval	tinyARC	tinyGSM8K: flexible match	tinyGSM8K: exact match	tiny Hellaswag	tinyMMLU	tiny TruthfulQA	tiny Winogrande
SLERP (t=0.3)	0,5793	0,6372	0,7347	0,7347	0,8235	0,6769	0,5585	0,744
SLERP (t=0.5)	0,5793	0,6536	0,778	0,778	0,8235	0,6646	0,5514	0,739
SLERP (t=0.7)	0,5427	0,6547	0,7851	0,7714	0,8268	0,6596	0,5409	0,7604
Breadcrumbs	0,5976	0,6586	0,8243	0,8243	0,8306	0,6646	0,5485	0,764
Breadcrumbs TIES	0,5549	0,6416	0,7645	0,7645	0,8306	0,6751	0,5418	0,7541

Analisi dei risultati



Conclusioni

- Superiorità dei Modelli Merged: Nella maggior parte degli scenari, le tecniche di merging superano in termini di performance il modello sottoposto a doppio fine-tuning.
- Stabilità delle Prestazioni: La quasi totalità dei metodi garantisce risultati paragonabili ai modelli addestrati su singoli dataset o al modello base, preservandone le capacità core.
- Recupero e Ottimizzazione:
 - Alcuni metodi, come Linear e DARE Linear, si dimostrano efficaci nel correggere i cali prestazionali derivanti da fine-tuning inefficienti.
 - Breadcrumbs si distingue come la soluzione più performante, essendo in grado non solo di recuperare le performance degradate, ma di migliorarle sensibilmente su tutti i task analizzati.