

Indice

1	Introduzione agli esperimenti	4
2	Prototypical Network con head lineare	6
2.1	Cosine Similarity	6
2.1.1	Esperimento 1 (n-way=5, n-shot=5, query=15, episodi=600, epoche=50)	7
2.1.2	Esperimento 2 (n-way=5, n-shot=3, query=15, episodi=600, epoche=50)	9
2.1.3	Esperimento 3 (n-way=5, n-shot=1, query=15, episodi=600, epoche=60)	11
2.1.4	Esperimento 4 (n-way=5, n-shot=5, query=15, episodi=1000, epoche=30)	13
2.1.5	Esperimento 5 (n-way=5, n-shot=5, query=15, episodi=1000, epoche=60)	15
2.2	Distanza Euclidea	17
2.2.1	Esperimento 1 (n-way=5, n-shot=5, query=15, episodi=600, epoche=50)	17
2.2.2	Esperimento 2 (n-way=5, n-shot=3, query=15, episodi=600, epoche=50)	19
2.2.3	Esperimento 3 (n-way=5, n-shot=1, query=15, episodi=600, epoche=60)	21
2.2.4	Esperimento 4 (n-way=5, n-shot=5, query=15, episodi=1000, epoche=30)	23
2.2.5	Esperimento 5 (n-way=5, n-shot=5, query=15, episodi=1000, epoche=60)	25
2.3	Confronto tra i risultati	27
3	Prototypical Networks con Projection Head Non Lineare (ReLU)	28
3.1	Cosine Similarity con ReLU	28
3.1.1	Esperimento 1 (n-way=5, n-shot=5, query=15, episodi=600, epoche=50)	29
3.1.2	Esperimento 2 (n-way=5, n-shot=3, query=15, episodi=600, epoche=50)	31
3.1.3	Esperimento 3 (n-way=5, n-shot=1, query=15, episodi=600, epoche=60)	33
3.1.4	Esperimento 4 (n-way=5, n-shot=5, query=15, episodi=1000, epoche=30)	35
3.1.5	Esperimento 5 (n-way=5, n-shot=5, query=15, episodi=1000, epoche=60)	37
3.2	Euclidean Distance con ReLU	39

3.2.1	Esperimento 1 (n-way=5, n-shot=5, query=15, episodi=600, epoche=50)	39
3.2.2	Esperimento 2 (n-way=5, n-shot=3, query=15, episodi=600, epoche=50)	41
3.2.3	Esperimento 3 (n-way=5, n-shot=1, query=15, episodi=600, epoche=60)	43
3.2.4	Esperimento 4 (n-way=5, n-shot=5, query=15, episodi=1000, epoche=30)	45
3.2.5	Esperimento 5 (n-way=5, n-shot=5, query=15, episodi=1000, epoche=60)	47
3.3	Confronto tra i risultati	49
4	Cosine similarity con Tanh	50
4.1	Esperimento 1 (n-way=5, n-shot=5, query=15, episodi=600, epoche=50)	51
4.2	Esperimento 2 (n-way=5, n-shot=3, query=15, episodi=600, epoche=50)	53
4.3	Esperimento 3 (n-way=5, n-shot=1, query=15, episodi=600, epoche=60)	55
4.4	Esperimento 4 (n-way=5, n-shot=5, query=15, episodi=1000, epoche=30)	57
4.5	Esperimento 5 (n-way=5, n-shot=5, query=15, episodi=1000, epoche=60)	59
5	Confronto risultati cosine tra Relu e Tanh	61
5.1	Esperimento 1	62
5.2	Esperimento 2	64
5.3	Esperimento 3	66
5.4	Esperimento 4	68
5.5	Esperimento 5	70
6	Ottimizzazione con SGD e Learning Rate Scheduler	72
6.1	ReLU	73
6.2	Esperimento 1 (ReLU)	73
6.3	Esperimento 2 (ReLU)	75
6.4	Esperimento 3 (ReLU)	77
6.5	Tanh	79
6.6	Esperimento 1 (Tanh)	79
6.7	Esperimento 2 (Tanh)	81
6.8	Esperimento 3 (Tanh)	83
6.9	Confronto risultati	84
6.9.1	Esperimento 1	86
6.9.2	Esperimento 2	89
6.9.3	Esperimento 3	91

7	Selezione del modello finale	94
7.1	Valutazione del modello finale	94
7.1.1	Valori sul Test	95
7.1.2	Confusion Matrix	95

1 Introduzione agli esperimenti

Nel seguente report vengono analizzati i risultati ottenuti applicando un modello *Prototypical Network* per la classificazione *few-shot* in modalità episodica, a partire da embedding pre-calcolati. L'obiettivo degli esperimenti è valutare il comportamento del modello al variare della configurazione episodica, della metrica di classificazione e della struttura della projection head.

I risultati sono organizzati secondo due criteri principali:

- la **metrica di classificazione** utilizzata, che sono:
 - **Cosine Similarity**, ovvero una misura di similarità tra vettori che valuta l'angolo tra di essi, ignorandone la norma:

$$\text{cosine}(x, y) = \frac{x \cdot y}{\|x\| \|y\|}, \quad x, y \in \mathbb{R}^d \quad (1)$$

La **cosine similarity** è particolarmente importante nei Prototypical Networks poichè consente di classificare un campione in base a quanto esso è direzionalmente simile al prototipo di una classe. Tra i principali vantaggi troviamo:

- * **Invarianza alla scala**
- * **Compatibilità con embedding normalizzati**
- * **Maggiore robustezza in scenari few-shot**
- **Distanza Euclidea**, che misura quanto due vettori sono lontani nello spazio delle feature in termini assoluti:

$$d_{\text{euc}}(x, y) = \|x - y\|_2, \quad x, y \in \mathbb{R}^d \quad (2)$$

Un limite della distanza euclidea è che, in presenza di feature normalizzate, risulta strettamente collegata alla **cosine similarity**, con la differenza che la prima è sensibile alla norma dei vettori. Questo può portare ai seguenti problemi:

- * Due embedding con la stessa direzione ma norma diversa possono risultare lontani.
 - * Può introdurre errori quando si lavora con feature pre-addestrate o non adeguatamente scalate
- la **struttura della projection head**, distinguendo tra una trasformazione puramente lineare e una trasformazione non lineare con funzione di attivazione ReLU o Tanh.

Gli esperimenti sono condotti utilizzando due differenti strategie di ottimizzazione:

- AdamW, impiegato nelle prime fasi sperimentali;
- SGD, introdotto successivamente in combinazione con un learning rate scheduler, per analizzare l'impatto di una dinamica di apprendimento più controllata sulla stabilità e sulla generalizzazione del modello.

I risultati sperimentali vengono presentati separatamente per ciascuna metrica di distanza e per ciascuna architettura della head, riportando l'andamento di accuracy, macro F1-score e loss sia in fase di addestramento che di validazione.

Per ciascuna configurazione, sono stati condotti **cinque esperimenti**, ciascuno caratterizzato da una diversa impostazione dei parametri episodici:

- **Esperimento 1:** 5-way, 5-shot, 15 query, 600 episodi, 50 epoche;
- **Esperimento 2:** 5-way, 3-shot, 15 query, 600 episodi, 50 epoche;
- **Esperimento 3:** 5-way, 1-shot, 15 query, 600 episodi, 60 epoche;
- **Esperimento 4:** 5-way, 3-shot, 15 query, 1000 episodi, 30 epoche;
- **Esperimento 5:** 5-way, 3-shot, 15 query, 1000 episodi, 60 epoche.

In questo modo è possibile analizzare:

- l'impatto del numero di esempi di support (*n-shot*);
- l'effetto del numero di episodi e delle epoche di addestramento;
- le differenze di comportamento tra cosine similarity e distanza euclidea;
- il contributo della non linearità introdotta dalle funzioni ReLU e Tanh nella projection head.

2 Prototypical Network con head lineare

In questo capitolo vengono analizzati i risultati ottenuti utilizzando la cosine similarity e la distanza euclidea come funzioni di confronto tra i prototipi di classe e i campioni di query in un modello Prototypical Network con projection head lineare.

A parità di architettura del backbone e di impostazione del training, l'analisi si concentra sull'impatto della metrica di distanza sulle prestazioni in scenari few-shot, in assenza di non linearità nella head.

Per ciascuna configurazione vengono svolti cinque esperimenti descritti nella [Sezione 1](#), e i risultati sono analizzati in termini di accuracy, F1-score e loss, considerando sia l'andamento durante l'addestramento sia le prestazioni finali in validazione.

2.1 Cosine Similarity

Add.	n-way	n-shot	Query	Episodi	Epoche	Train Acc	Val Acc	Train F1	Val F1	Gap
1	5	5	15	600	50	≈ 0.763	≈ 0.746	≈ 0.761	≈ 0.742	$\approx 1.74\%$
2	5	3	15	600	50	≈ 0.729	≈ 0.717	≈ 0.725	≈ 0.714	$\approx 1.21\%$
3	5	1	15	600	60	≈ 0.614	≈ 0.592	≈ 0.601	≈ 0.580	$\approx 2.16\%$
4	5	5	15	1000	30	≈ 0.752	≈ 0.745	≈ 0.750	≈ 0.742	$\approx 0.72\%$
5	5	5	15	1000	60	≈ 0.755	≈ 0.756	≈ 0.753	≈ 0.754	$\approx 0.16\%$

Tabella 1: Configurazione di addestramento del modello Prototypical Network (distanza cosine, learning rate 1×10^{-3}).

2.1.1 Esperimento 1 (n-way=5, n-shot=5, query=15, episodi=600, epoche=50)

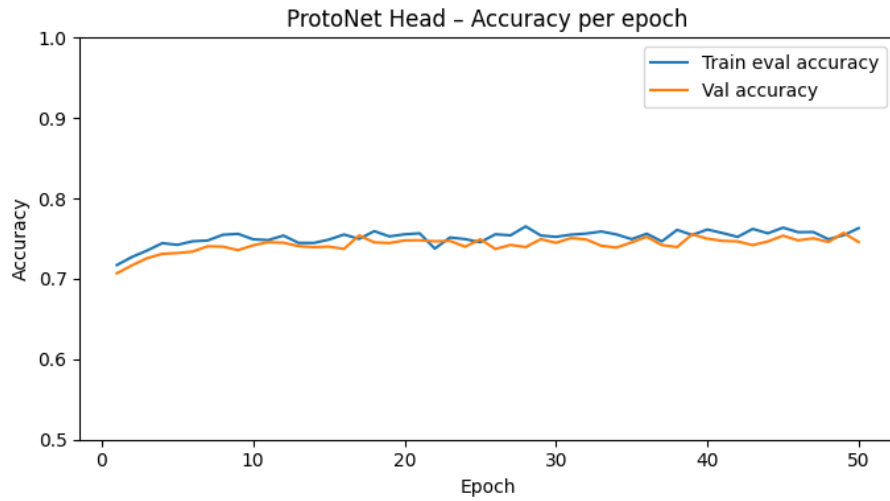


Figura 1: Esperimento 1: andamento dell'accuracy (train vs validation) durante l'addestramento.

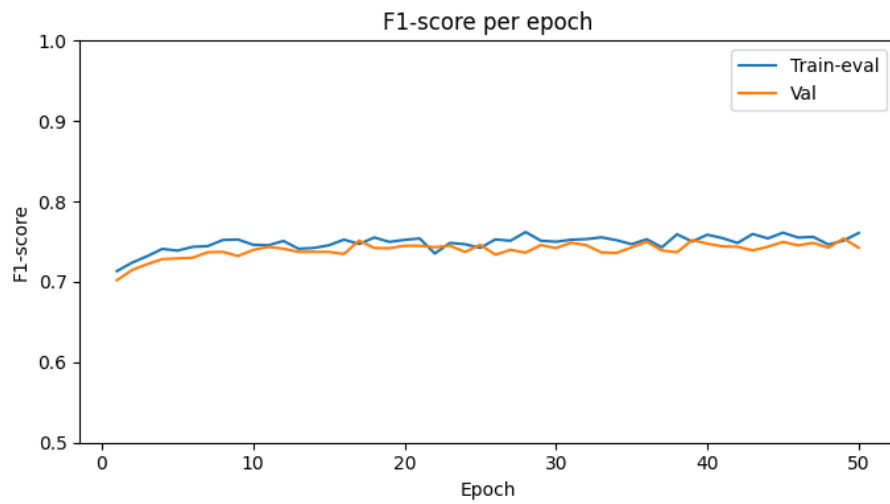


Figura 2: Esperimento 1: andamento del macro F1-score (train vs validation) durante l'addestramento.

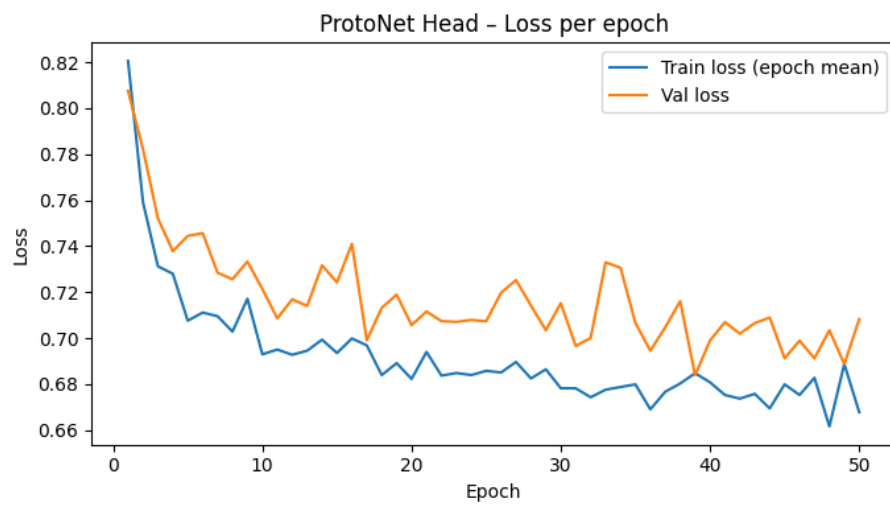


Figura 3: Esperimento 1: andamento della loss (train vs validation) durante l'addestramento.

[Torna alla tabella dei risultati](#)

2.1.2 Esperimento 2 (n-way=5, n-shot=3, query=15, episodi=600, epoche=50)

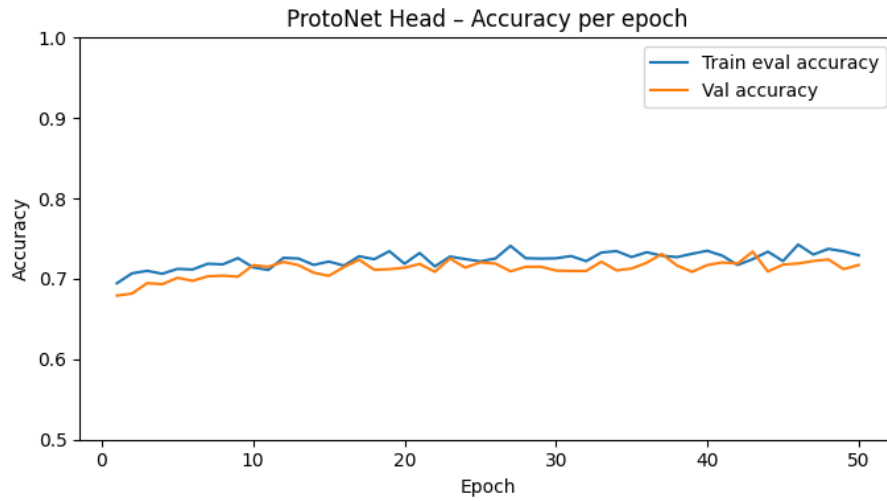


Figura 4: Esperimento 2: andamento dell'accuracy (train vs validation) durante l'addestramento.

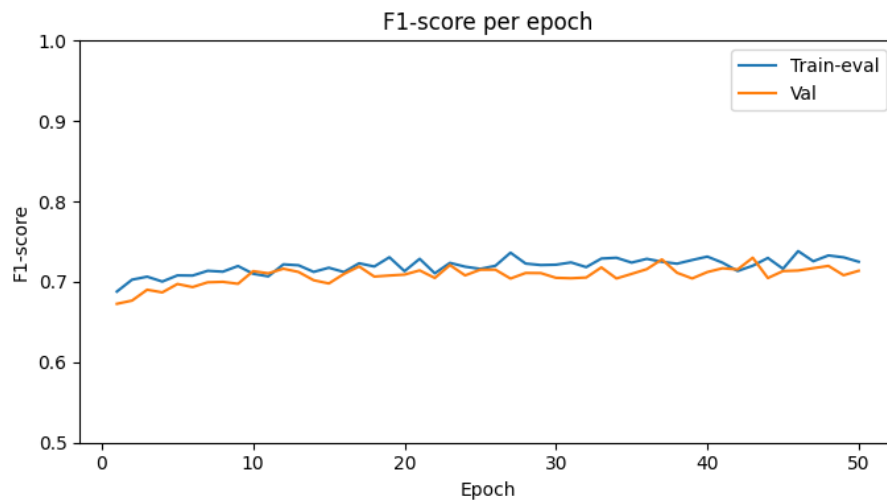


Figura 5: Esperimento 2: andamento del macro F1-score (train vs validation) durante l'addestramento.

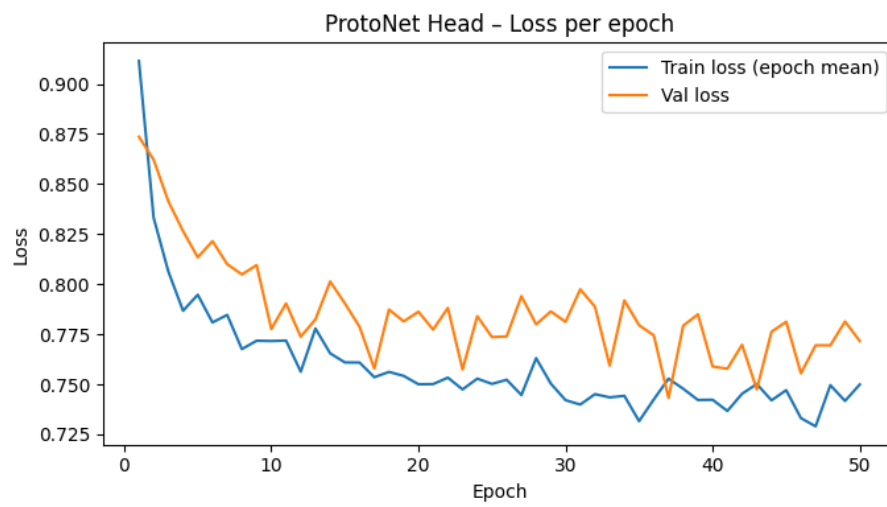


Figura 6: Esperimento 2: andamento della loss (train vs validation) durante l'addestramento.

[Torna alla tabella dei risultati](#)

2.1.3 Esperimento 3 (n-way=5, n-shot=1, query=15, episodi=600, epoche=60)

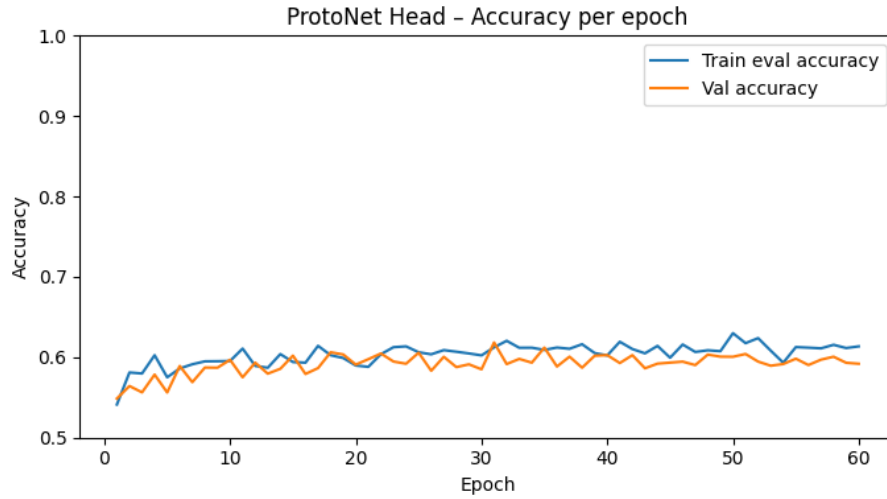


Figura 7: Esperimento 3: andamento dell'accuracy (train vs validation) durante l'addestramento.

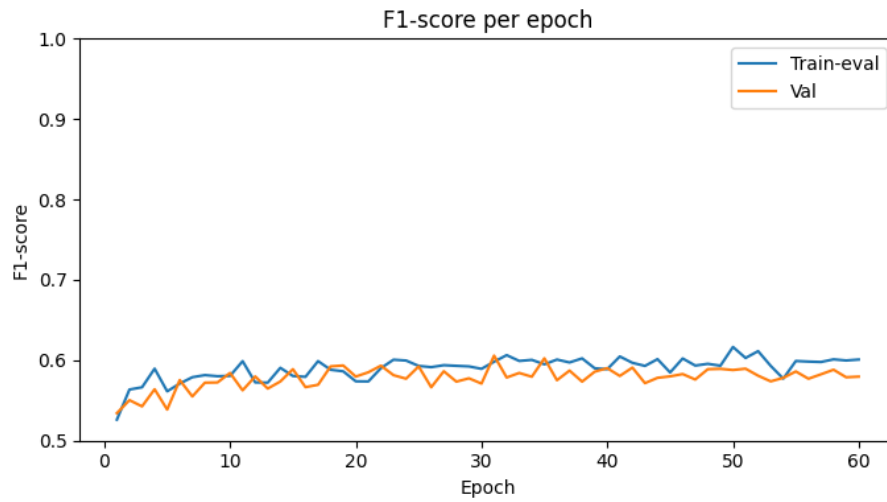


Figura 8: Esperimento 3: andamento del macro F1-score (train vs validation) durante l'addestramento.

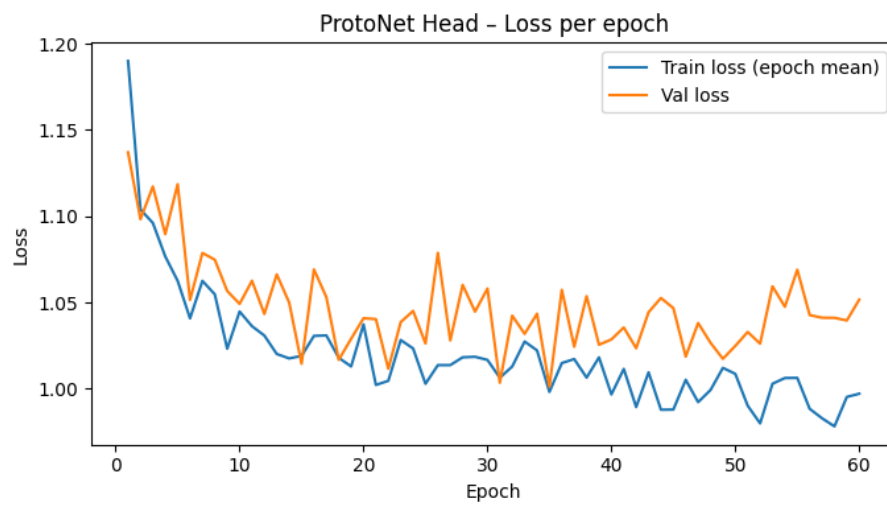


Figura 9: Esperimento 3: andamento della loss (train vs validation) durante l'addestramento.

[Torna alla tabella dei risultati](#)

2.1.4 Esperimento 4 (n-way=5, n-shot=5, query=15, episodi=1000, epoche=30)

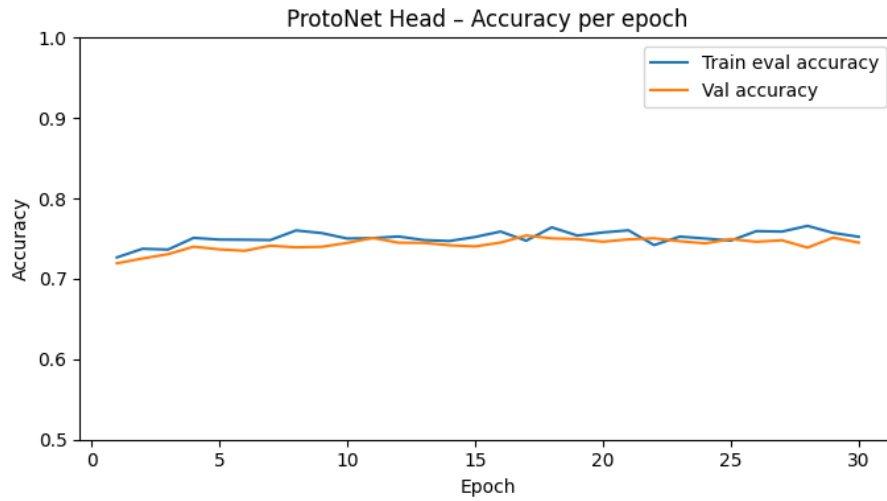


Figura 10: Esperimento 4: andamento dell'accuracy (train vs validation) durante l'addestramento.

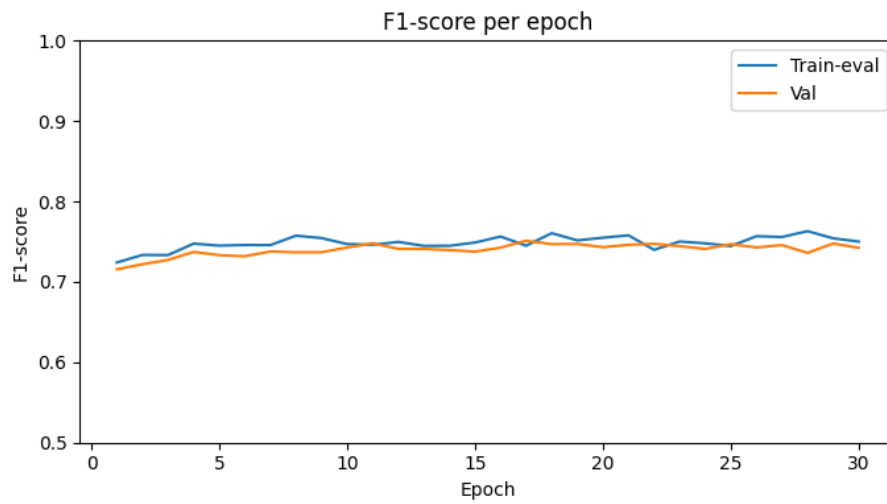


Figura 11: Esperimento 4: andamento del macro F1-score (train vs validation) durante l'addestramento.

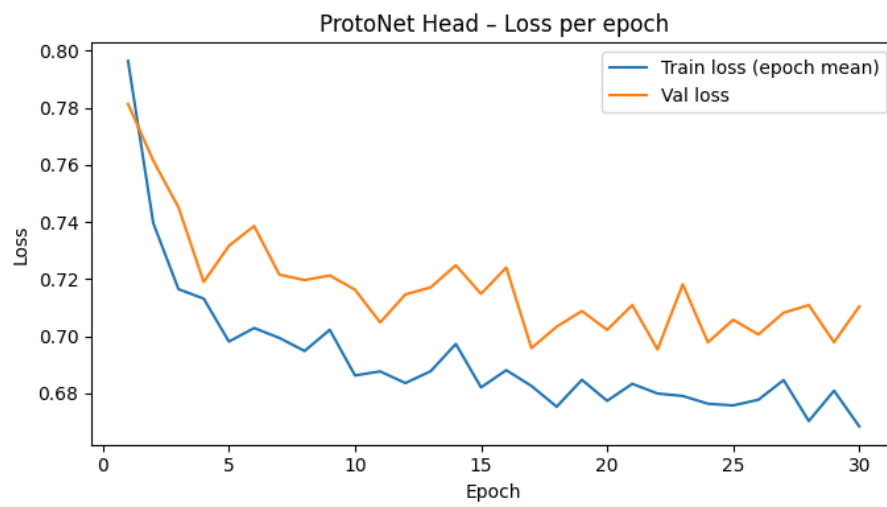


Figura 12: Esperimento 4: andamento della loss (train vs validation) durante l'addestramento.

[Torna alla tabella dei risultati](#)

2.1.5 Esperimento 5 (n-way=5, n-shot=5, query=15, episodi=1000, epoche=60)

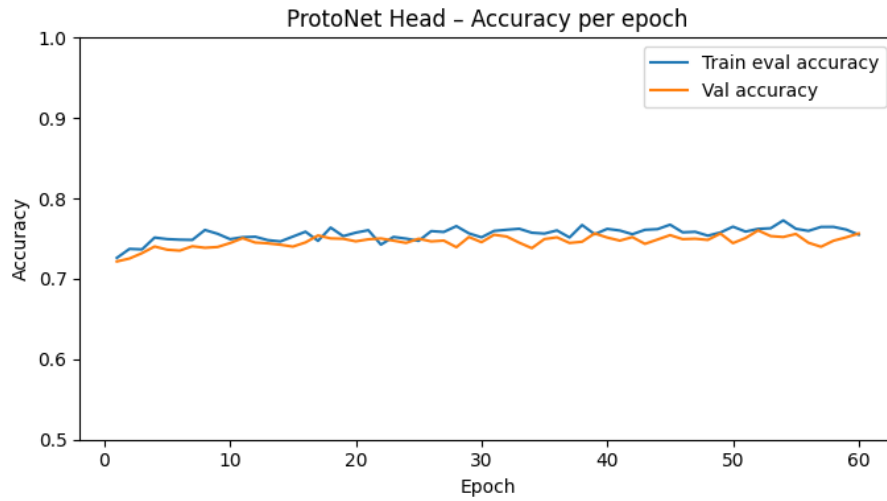


Figura 13: Esperimento 5: andamento dell'accuracy (train vs validation) durante l'addestramento.

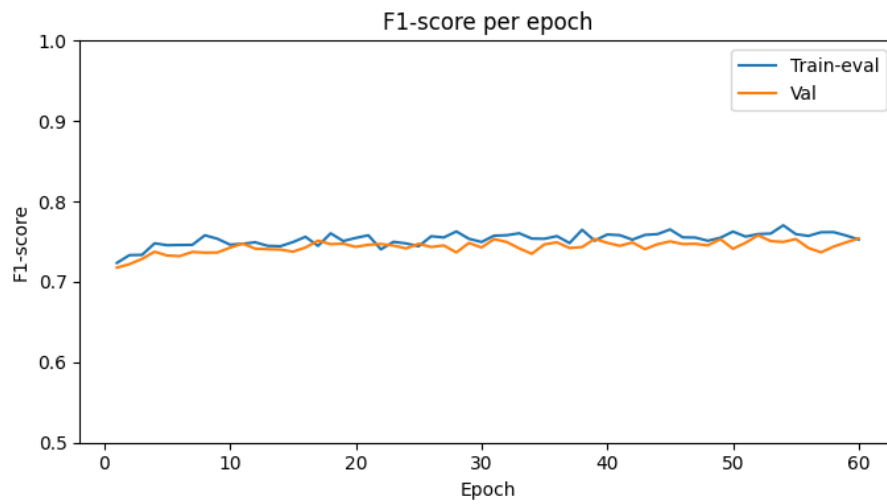


Figura 14: Esperimento 5: andamento del macro F1-score (train vs validation) durante l'addestramento.

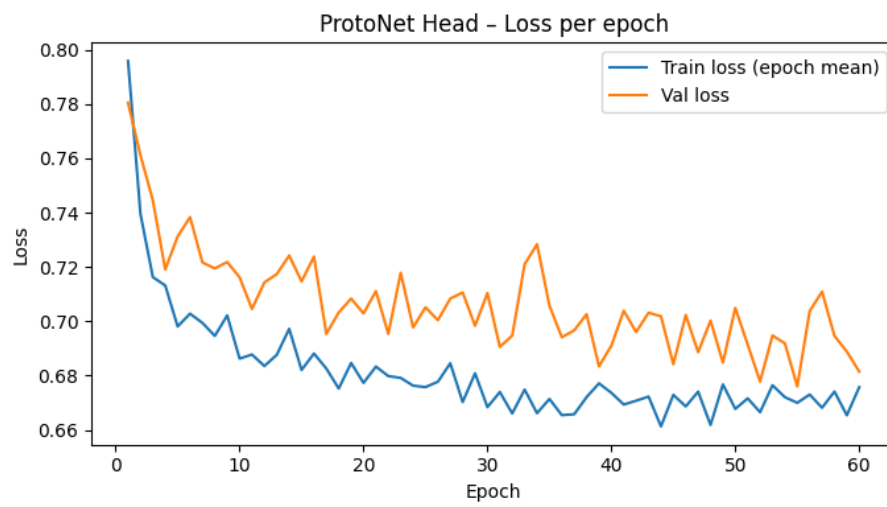


Figura 15: Esperimento 5: andamento della loss (train vs validation) durante l'addestramento.

[Torna alla tabella dei risultati](#)

2.2 Distanza Euclidea

Add.	n-way	n-shot	Query	Episodi	Epoche	Train Acc	Val Acc	Train F1	Val F1	Gap
1	5	5	15	600	50	≈ 0.737	≈ 0.724	≈ 0.735	≈ 0.721	$\approx 1.29\%$
2	5	3	15	600	50	≈ 0.702	≈ 0.686	≈ 0.698	≈ 0.682	$\approx 1.59\%$
3	5	1	15	600	60	≈ 0.529	≈ 0.517	≈ 0.510	≈ 0.503	$\approx 1.18\%$
4	5	5	15	1000	30	≈ 0.724	≈ 0.722	≈ 0.722	≈ 0.719	$\approx 0.19\%$
5	5	5	15	1000	60	≈ 0.722	≈ 0.724	≈ 0.719	≈ 0.721	$\approx 0.15\%$

Tabella 2: Configurazione di addestramento del modello Prototypical Network (distanza euclidea, learning rate 1×10^{-3}).

2.2.1 Esperimento 1 (n-way=5, n-shot=5, query=15, episodi=600, epoche=50)

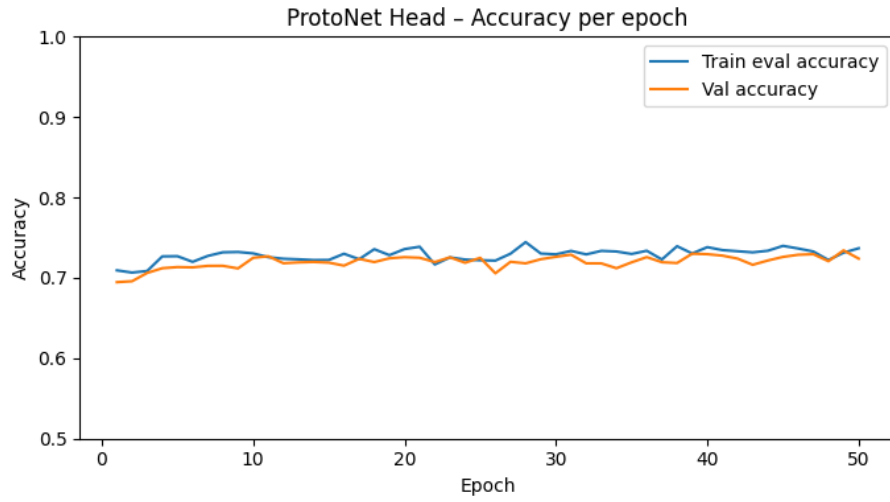


Figura 16: Esperimento EC1: andamento dell'accuracy (train vs validation) durante l'addestramento.

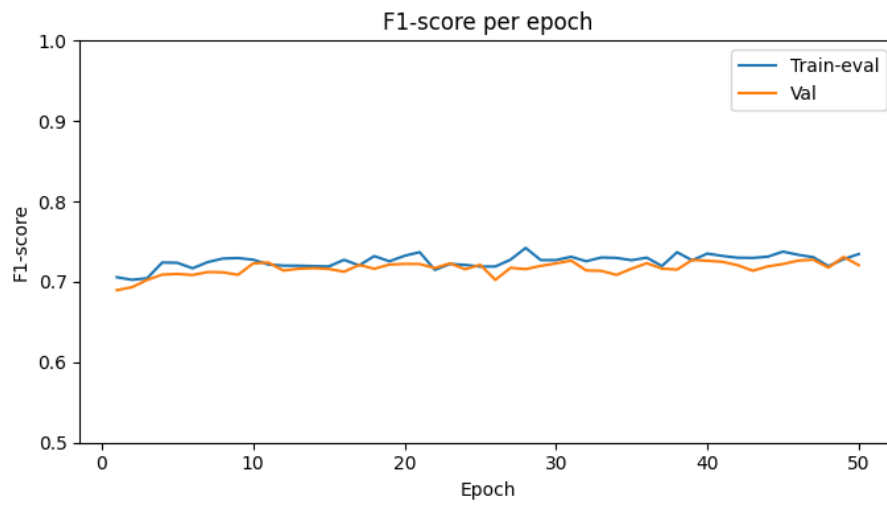


Figura 17: Esperimento EC1: andamento del macro F1-score (train vs validation) durante l'addestramento.

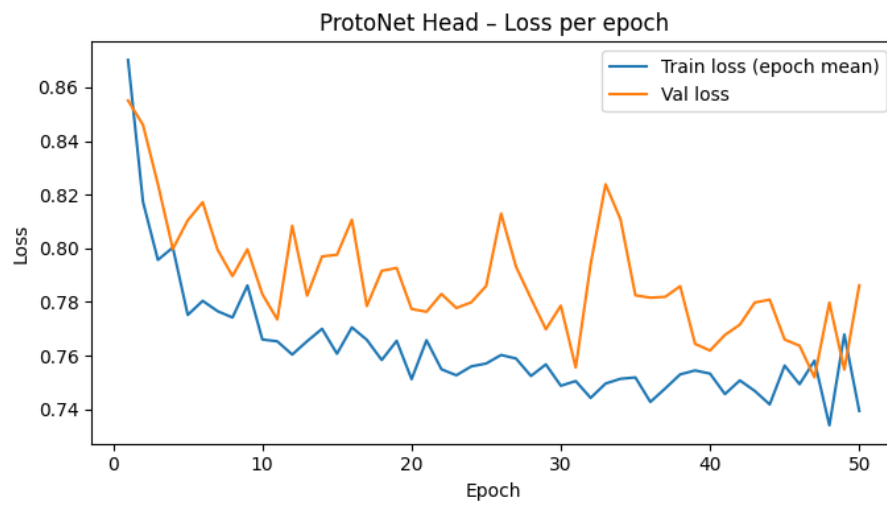


Figura 18: Esperimento EC1: andamento della loss (train vs validation) durante l'addestramento.

[Torna alla tabella dei risultati](#)

2.2.2 Esperimento 2 (n-way=5, n-shot=3, query=15, episodi=600, epoche=50)

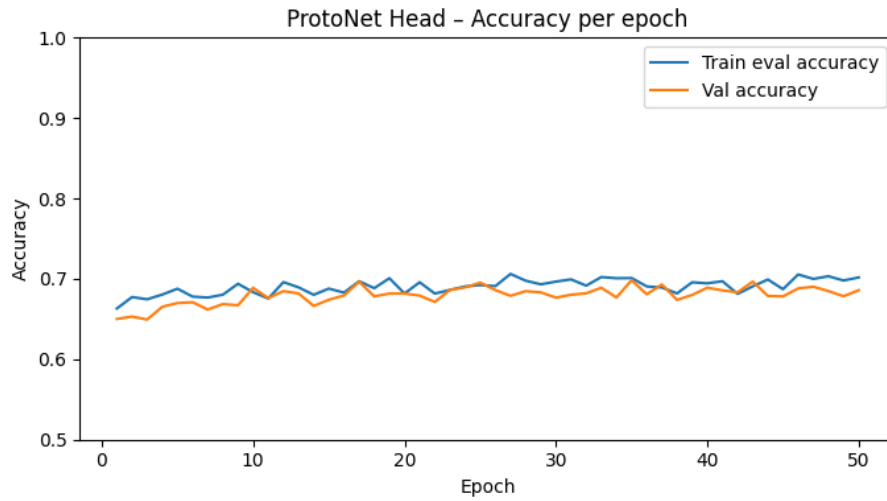


Figura 19: Esperimento EC2: andamento dell'accuracy (train vs validation) durante l'addestramento.

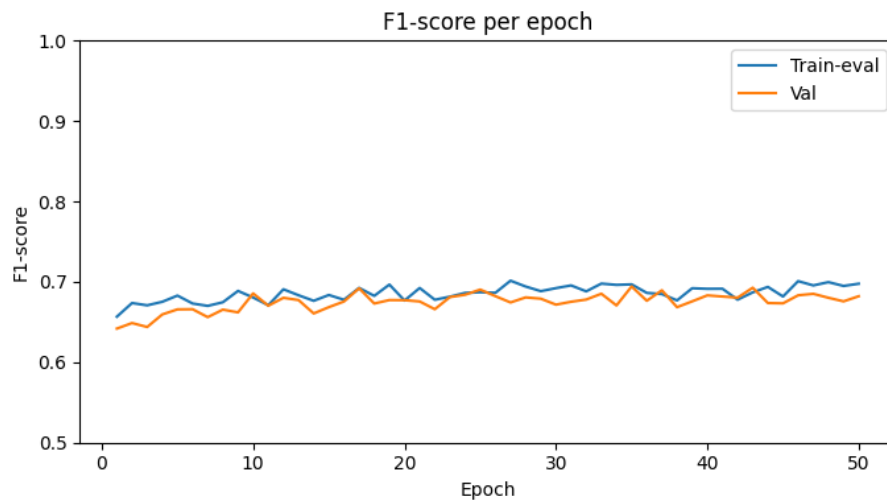


Figura 20: Esperimento EC2: andamento del macro F1-score (train vs validation) durante l'addestramento.

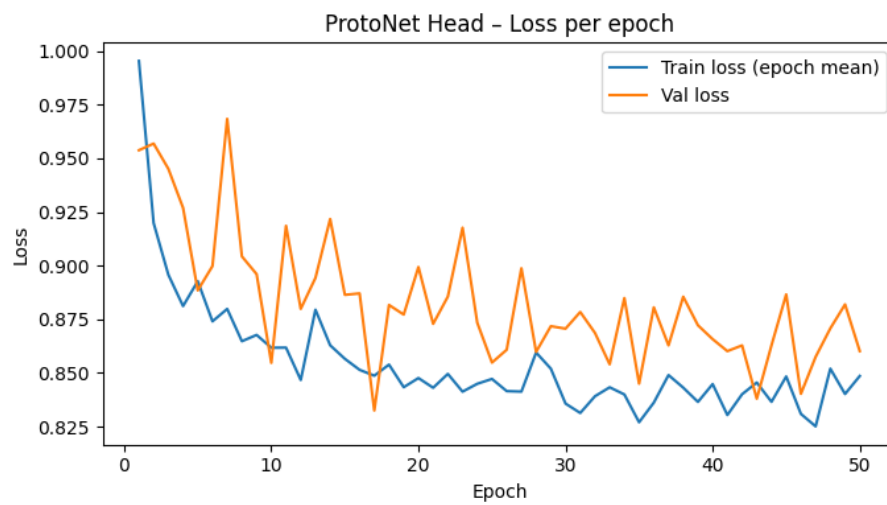


Figura 21: Esperimento EC2: andamento della loss (train vs validation) durante l'addestramento.

[*Torna alla tabella dei risultati*](#)

2.2.3 Esperimento 3 (n-way=5, n-shot=1, query=15, episodi=600, epoche=60)

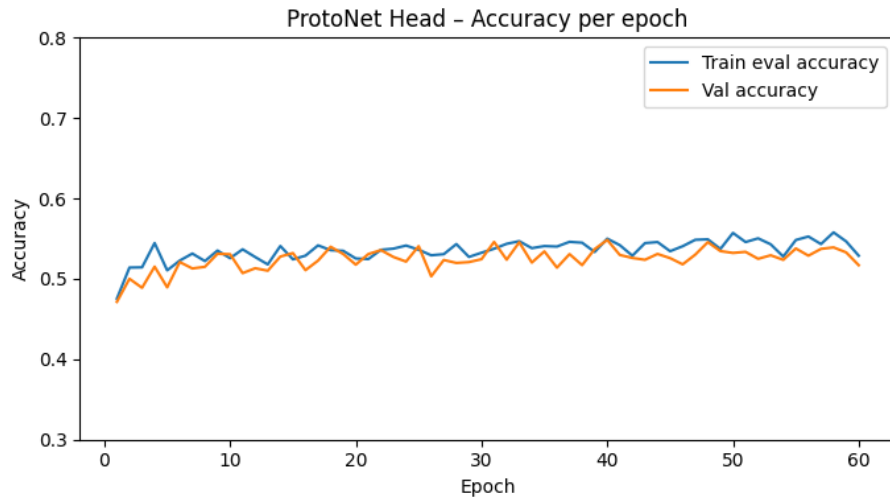


Figura 22: Esperimento EC3: andamento dell'accuracy (train vs validation) durante l'addestramento.

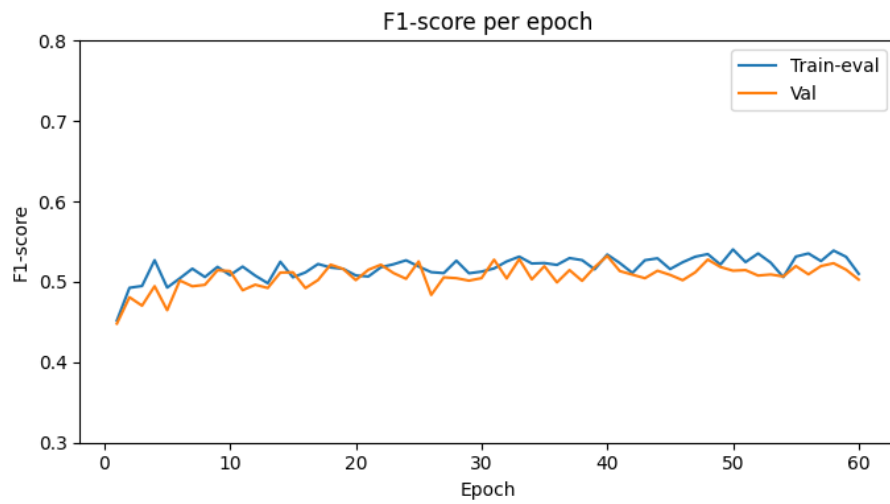


Figura 23: Esperimento EC3: andamento del macro F1-score (train vs validation) durante l'addestramento.

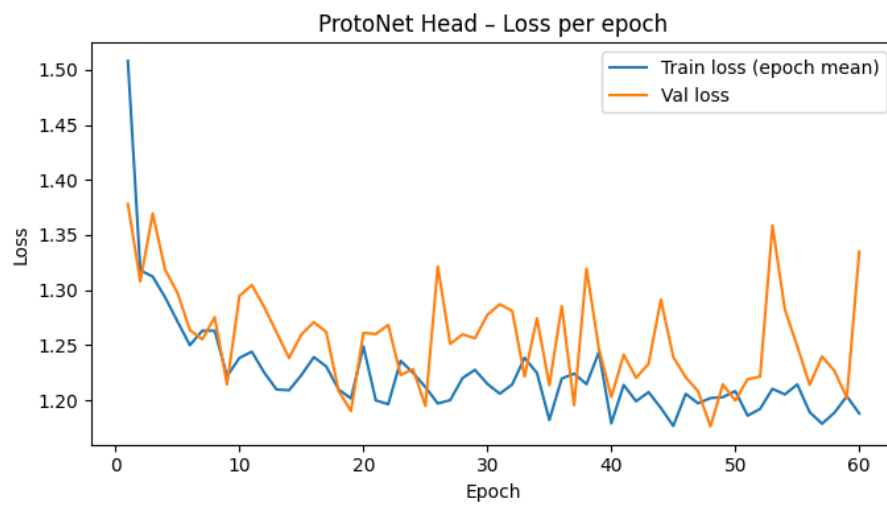


Figura 24: Esperimento EC3: andamento della loss (train vs validation) durante l'addestramento.

[Torna alla tabella dei risultati](#)

2.2.4 Esperimento 4 (n-way=5, n-shot=5, query=15, episodi=1000, epoche=30)

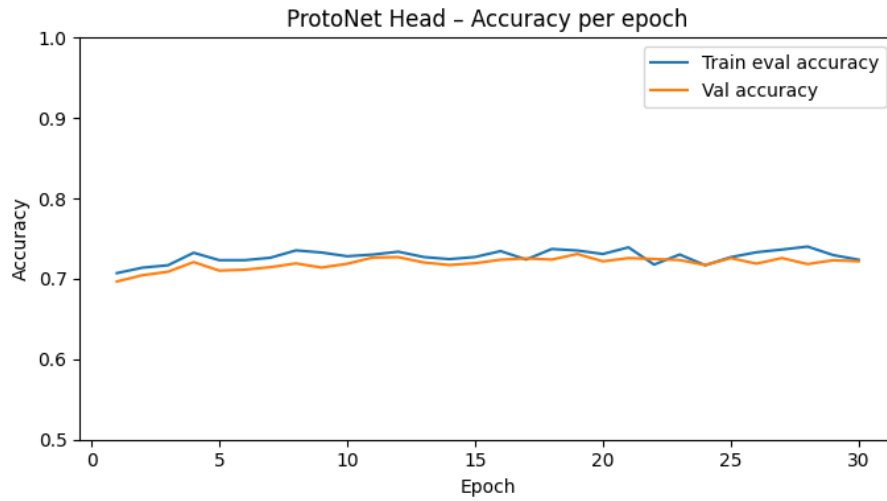


Figura 25: Esperimento EC4: andamento dell'accuracy (train vs validation) durante l'addestramento.

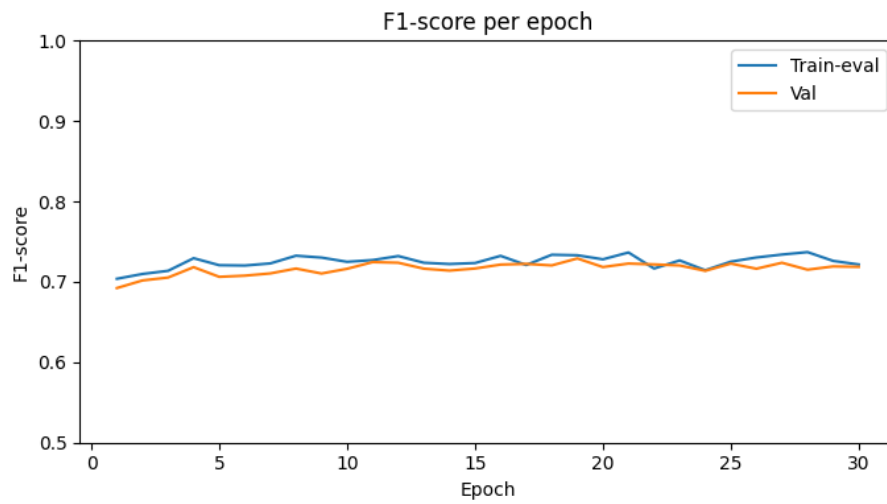


Figura 26: Esperimento EC4: andamento del macro F1-score (train vs validation) durante l'addestramento.

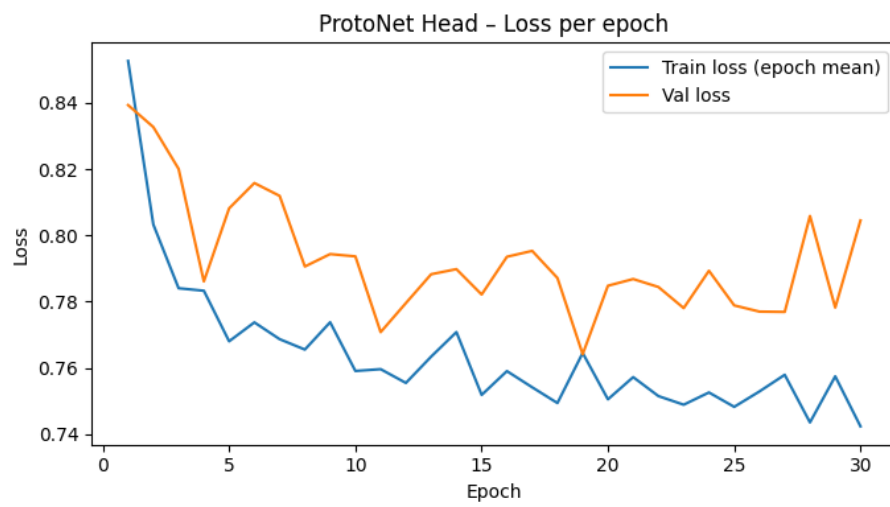


Figura 27: Esperimento EC4: andamento della loss (train vs validation) durante l'addestramento.

[Torna alla tabella dei risultati](#)

2.2.5 Esperimento 5 (n-way=5, n-shot=5, query=15, episodi=1000, epoche=60)

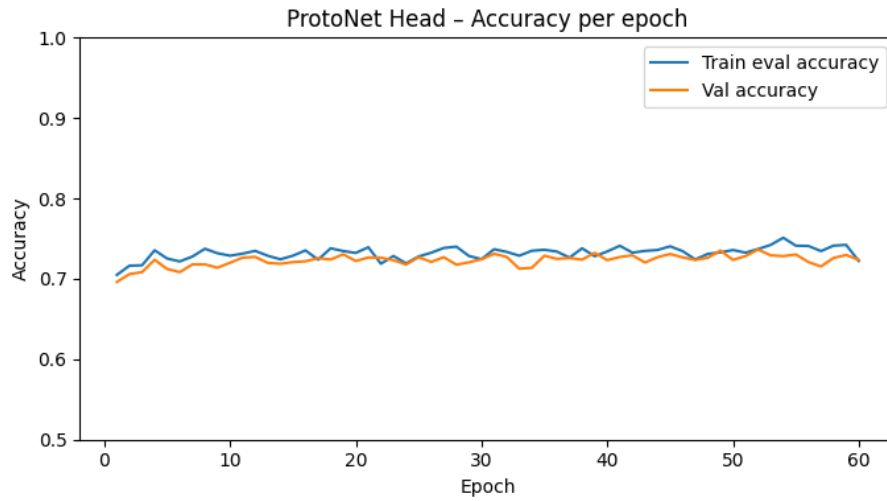


Figura 28: EsperimentoE5: andamento dell'accuracy (train vs validation) durante l'addestramento.

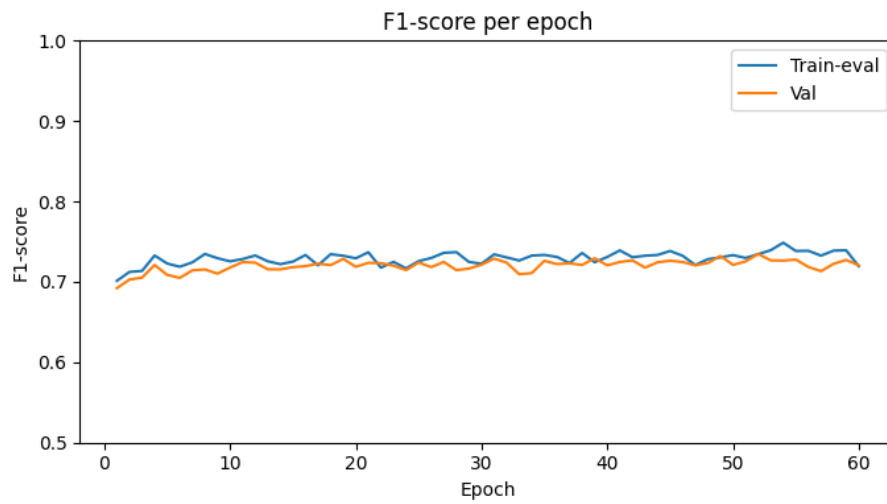


Figura 29: EsperimentoE5: andamento del macro F1-score (train vs validation) durante l'addestramento.

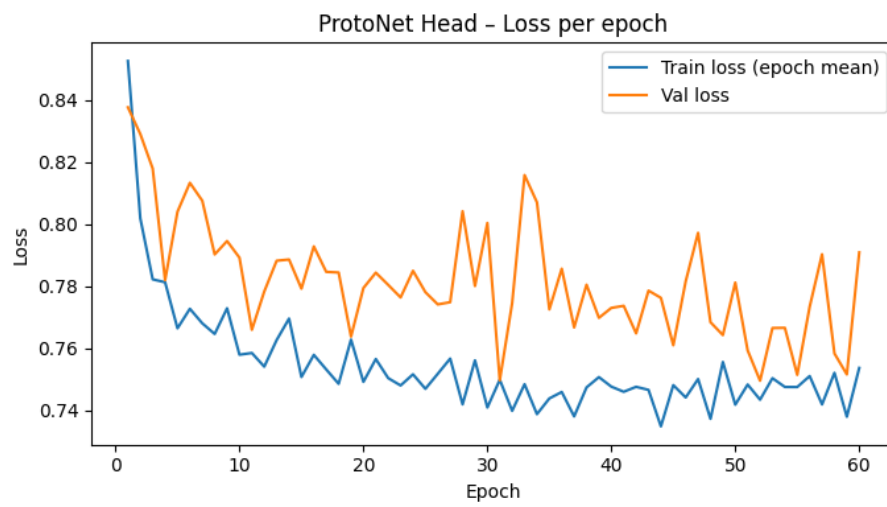


Figura 30: EsperimentoE5: andamento della loss (train vs validation) durante l'addestramento.

[Torna alla tabella dei risultati](#)

2.3 Confronto tra i risultati

Exp	n-shot	Episodi	Epoche	Val Acc (Cosine / Euclidean)	Val F1 (Cosine / Euclidean)	Gap migliore
1	5	600	50	0.746 / 0.724	0.742 / 0.721	Euclidean
2	3	600	50	0.717 / 0.686	0.714 / 0.682	Cosine
3	1	600	60	0.592 / 0.517	0.580 / 0.503	Euclidean
4	5	1000	30	0.745 / 0.722	0.742 / 0.719	Euclidean
5	5	1000	60	0.756 / 0.724	0.754 / 0.721	Euclidean

Tabella 3: Confronto tra distanza cosine e distanza euclidea per Prototypical Network (solo metriche di validazione). In *Gap migliore* è indicata la metrica con gap train-validation più basso.

Dal confronto dei risultati riportati in Tabella emerge che, in tutti gli esperimenti condotti, la cosine similarity ottiene valori marginalmente superiori sia in termini di accuratezza di validazione sia di F1-score rispetto alla distanza euclidea. Questo comportamento è consistente al variare del numero di shot e del numero di episodi di addestramento, suggerendo una maggiore stabilità della cosine similarity nel contesto few-shot quando si utilizza esclusivamente un layer lineare (nn.Linear).

La distanza euclidea, pur ottenendo prestazioni complessivamente inferiori in termini assoluti, mostra nella maggior parte degli esperimenti (ad eccezione dell'Esperimento 2) un gap migliore tra training e validation, indicando una maggiore capacità di generalizzazione. Infine, il minor divario tra le prestazioni di addestramento e validazione osservato con la distanza euclidea può essere ricondotto alla natura stessa della metrica, che risulta sensibile sia alla direzione sia alla magnitudine degli embedding. Questo comportamento introduce un vincolo più restrittivo nello spazio delle rappresentazioni, limitando la possibilità per il modello basato su un semplice layer lineare di adattarsi eccessivamente ai dati di training. Di conseguenza, sebbene le prestazioni assolute risultino leggermente inferiori rispetto alla cosine similarity, la distanza euclidea favorisce una generalizzazione più stabile e un gap train-validation più contenuto.

3 Prototypical Networks con Projection Head Non Lineare (ReLU)

In questo capitolo vengono analizzati i risultati ottenuti utilizzando la cosine similarity e la distanza euclidea come funzione di confronto tra i prototipi di classe e i campioni di query all'interno di un modello Prototypical Network dotato di una projection head non lineare con funzione di attivazione ReLU. A parità di architettura del backbone e di impostazione generale del training, l'obiettivo è valutare il contributo della non linearità introdotta nella projection head e il suo impatto sulle prestazioni in scenari few-shot, in combinazione con diverse metriche di distanza.

Dal punto di vista architetturale, il primo layer lineare produce una proiezione degli embedding in uno spazio latente, mentre la funzione ReLU introduce una non-linearità che spegne selettivamente le attivazioni negative, generando rappresentazioni sparse. Il secondo layer lineare ricombina quindi esclusivamente le componenti positive rimaste attive, aumentando la capacità espressiva del modello. Per ciascuna configurazione vengono svolti cinque esperimenti descritti nella [Sezione 1](#), e i risultati sono analizzati in termini di accuracy, F1-score e loss, considerando sia l'andamento durante l'addestramento sia le prestazioni finali in validazione.

3.1 Cosine Similarity con ReLU

Add.	n-way	n-shot	Query	Episodi	Epoche	Train Acc	Val Acc	Train F1	Val F1	Gap
1	5	5	15	600	50	≈ 0.822	≈ 0.806	≈ 0.821	≈ 0.805	$\approx 1.54\%$
2	5	3	15	600	50	≈ 0.811	≈ 0.785	≈ 0.809	≈ 0.783	$\approx 2.55\%$
3	5	1	15	600	60	≈ 0.704	≈ 0.668	≈ 0.695	≈ 0.656	$\approx 3.61\%$
4	5	5	15	1000	30	≈ 0.818	≈ 0.805	≈ 0.817	≈ 0.804	$\approx 1.26\%$
5	5	5	15	1000	60	≈ 0.826	≈ 0.813	≈ 0.825	≈ 0.812	$\approx 1.23\%$

Tabella 4: Configurazione di addestramento del modello Prototypical Network con head MLP (Linear-ReLU-Linear), distanza cosine, learning rate 1×10^{-3} .

3.1.1 Esperimento 1 (n-way=5, n-shot=5, query=15, episodi=600, epoche=50)

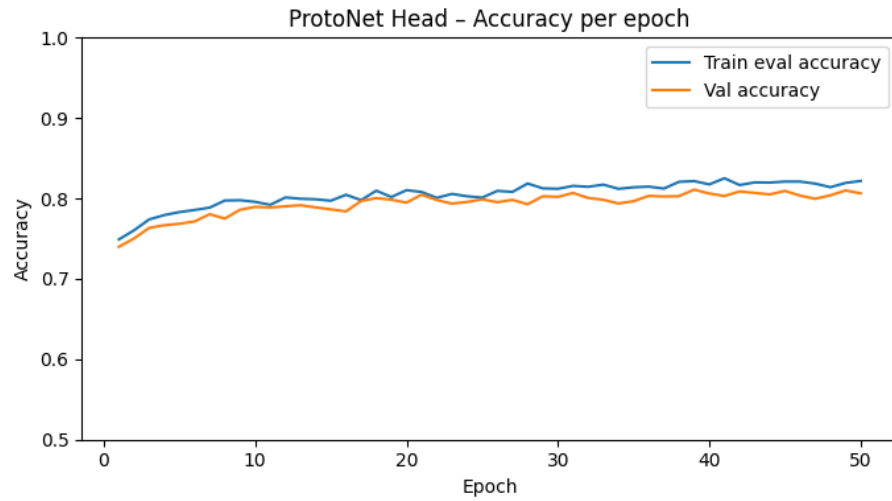


Figura 31: Esperimento 1: andamento dell'accuracy (train vs validation) durante l'addestramento.

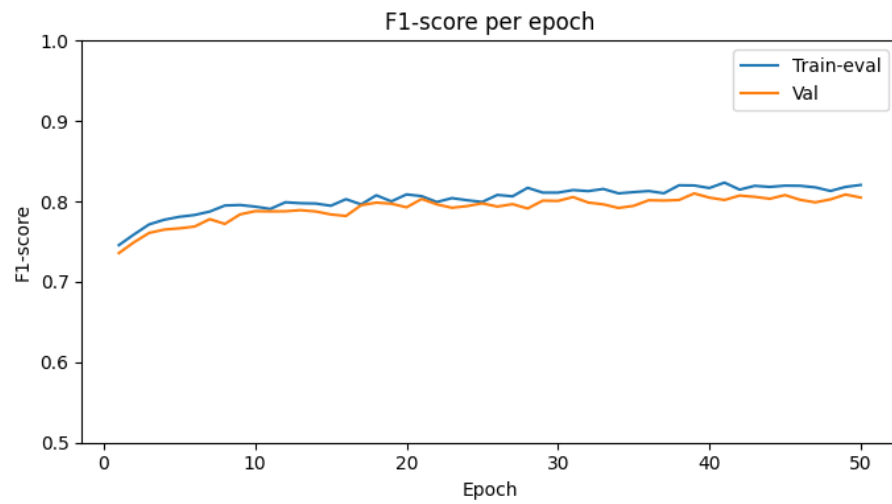


Figura 32: Esperimento 1: andamento del macro F1-score (train vs validation) durante l'addestramento.

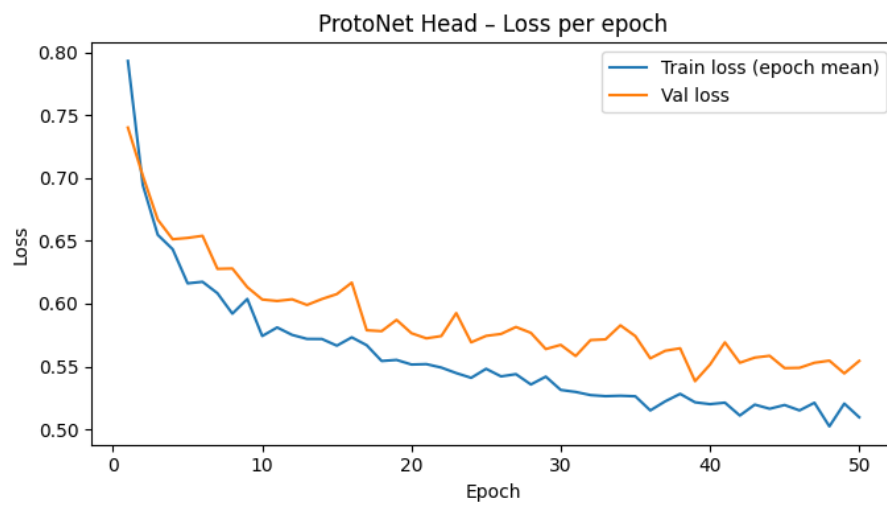


Figura 33: Esperimento 1: andamento della loss (train vs validation) durante l'addestramento.

[*Torna alla tabella dei risultati*](#)

3.1.2 Esperimento 2 (n-way=5, n-shot=3, query=15, episodi=600, epoche=50)

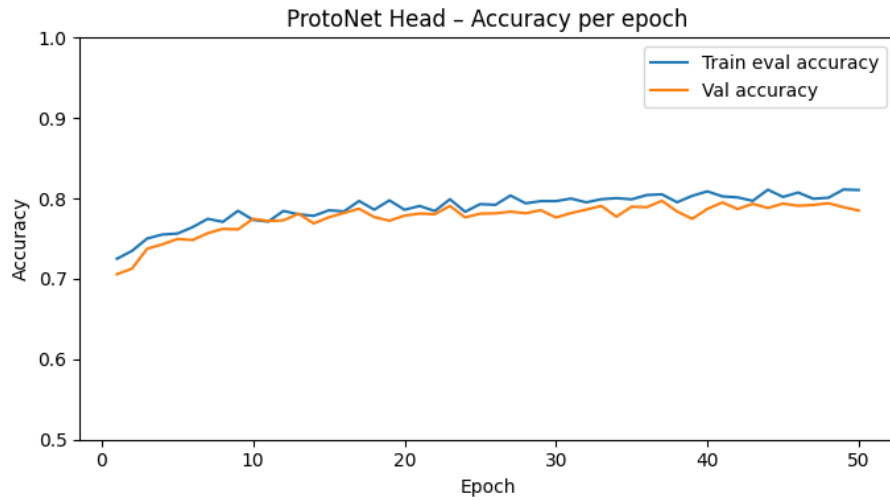


Figura 34: Esperimento 2: andamento dell'accuracy (train vs validation) durante l'addestramento.

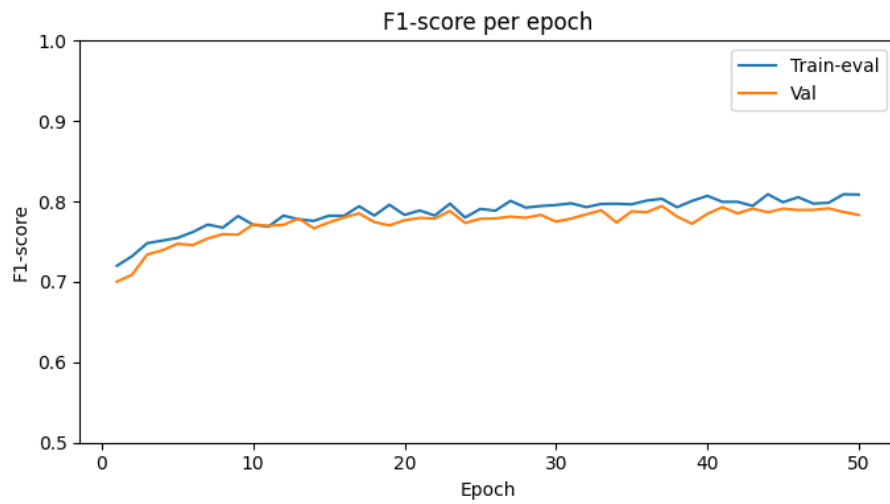


Figura 35: Esperimento 2: andamento del macro F1-score (train vs validation) durante l'addestramento.

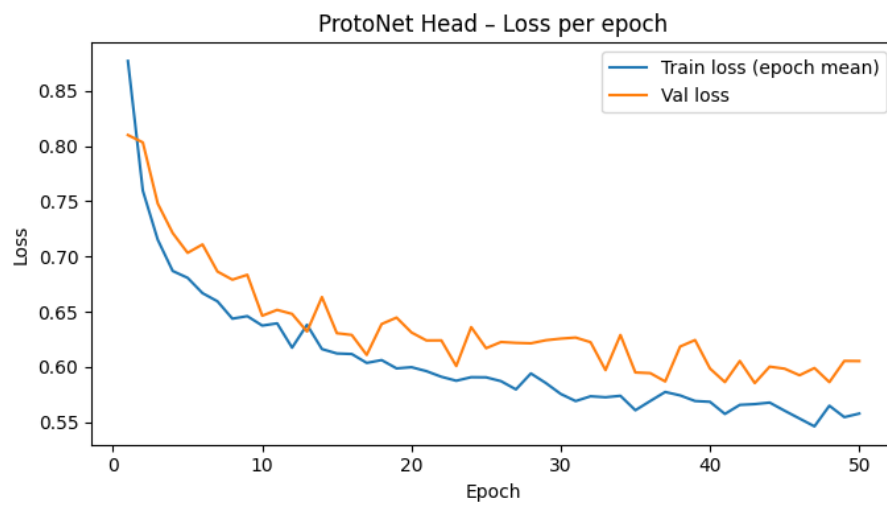


Figura 36: Esperimento 2: andamento della loss (train vs validation) durante l'addestramento.

[Torna alla tabella dei risultati](#)

3.1.3 Esperimento 3 (n-way=5, n-shot=1, query=15, episodi=600, epoche=60)

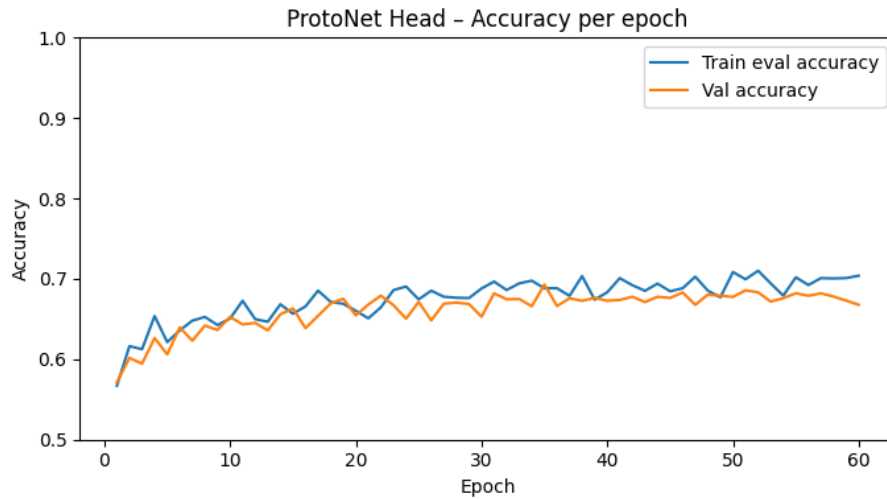


Figura 37: Esperimento 3: andamento dell'accuracy (train vs validation) durante l'addestramento.

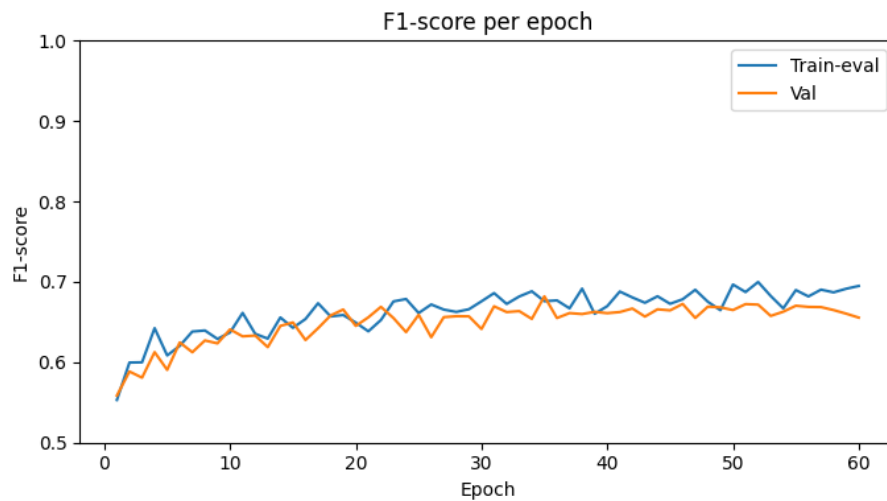


Figura 38: Esperimento 3: andamento del macro F1-score (train vs validation) durante l'addestramento.

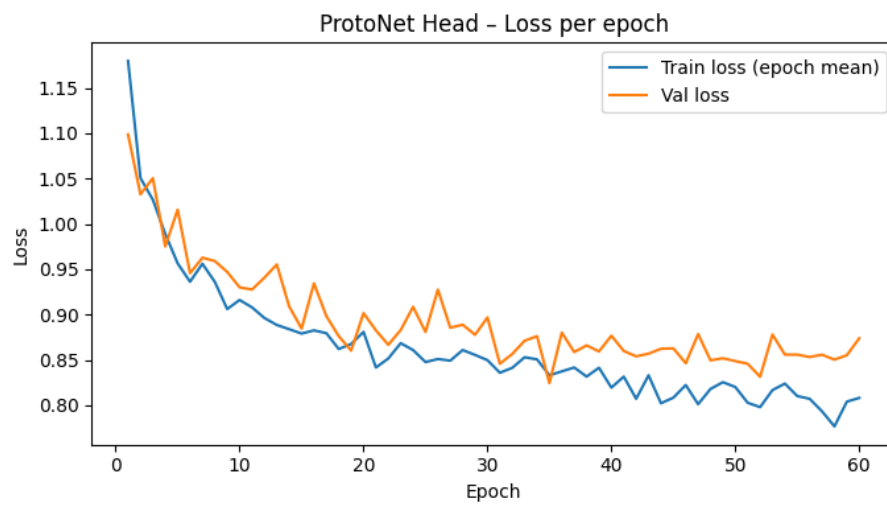


Figura 39: Esperimento 3: andamento della loss (train vs validation) durante l'addestramento.

[Torna alla tabella dei risultati](#)

3.1.4 Esperimento 4 (n-way=5, n-shot=5, query=15, episodi=1000, epoche=30)

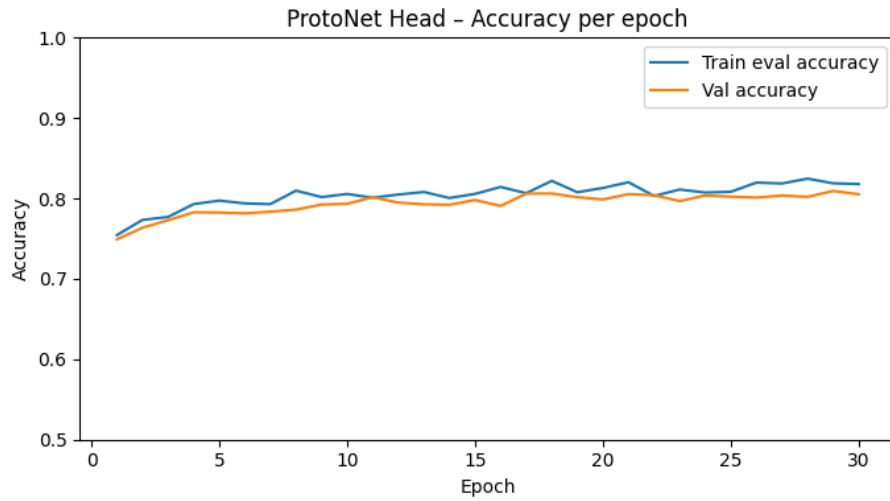


Figura 40: Esperimento 4: andamento dell'accuracy (train vs validation) durante l'addestramento.

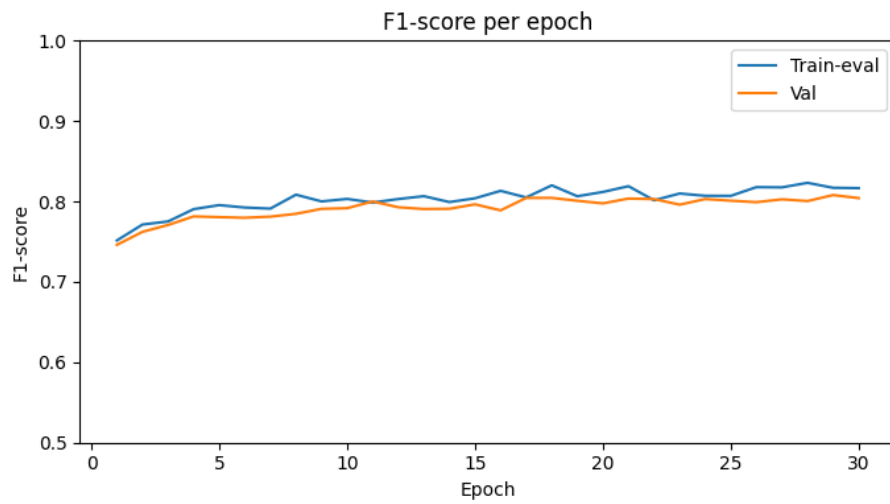


Figura 41: Esperimento 4: andamento del macro F1-score (train vs validation) durante l'addestramento.

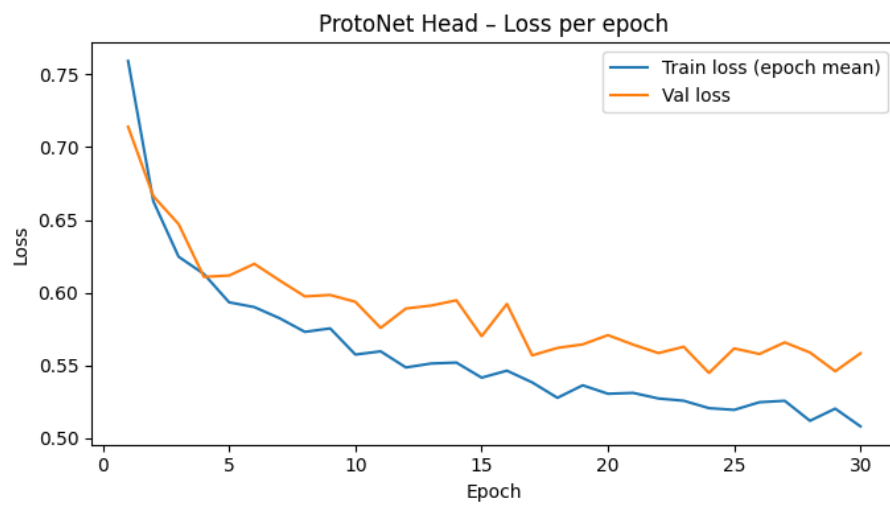


Figura 42: Esperimento 4: andamento della loss (train vs validation) durante l'addestramento.

[Torna alla tabella dei risultati](#)

3.1.5 Esperimento 5 (n-way=5, n-shot=5, query=15, episodi=1000, epoche=60)

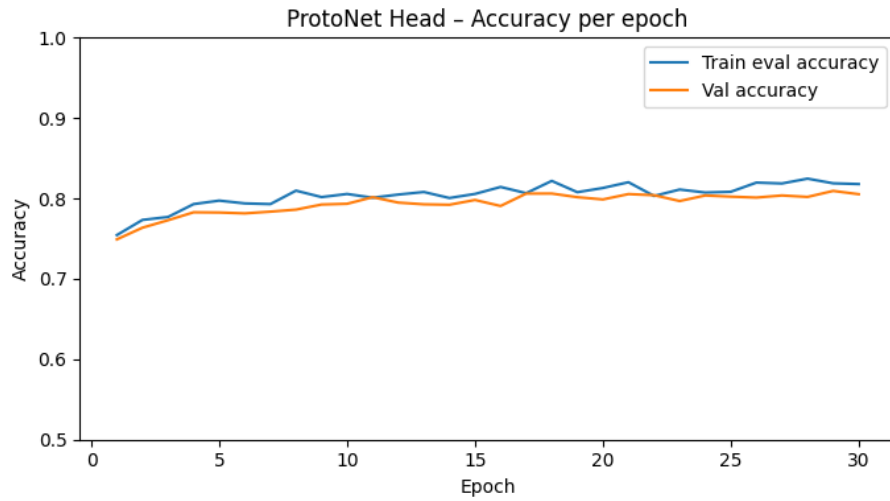


Figura 43: Esperimento 5: andamento dell'accuracy (train vs validation) durante l'addestramento.

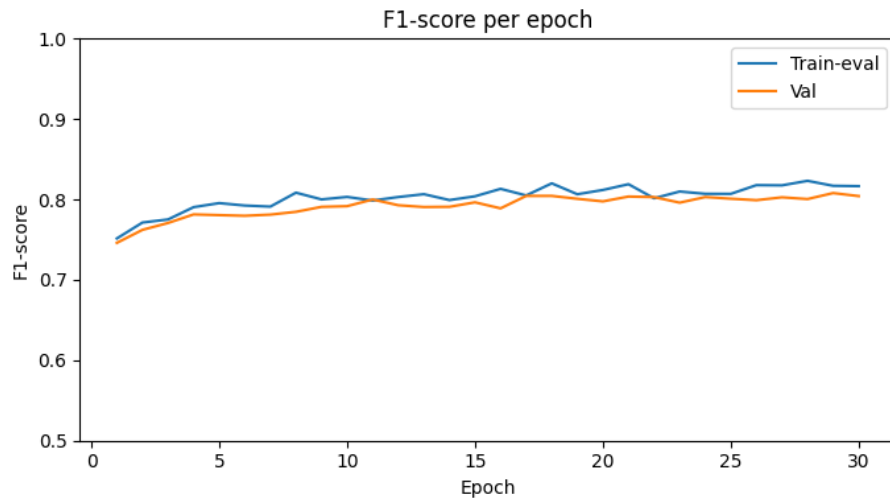


Figura 44: Esperimento 5: andamento del macro F1-score (train vs validation) durante l'addestramento.

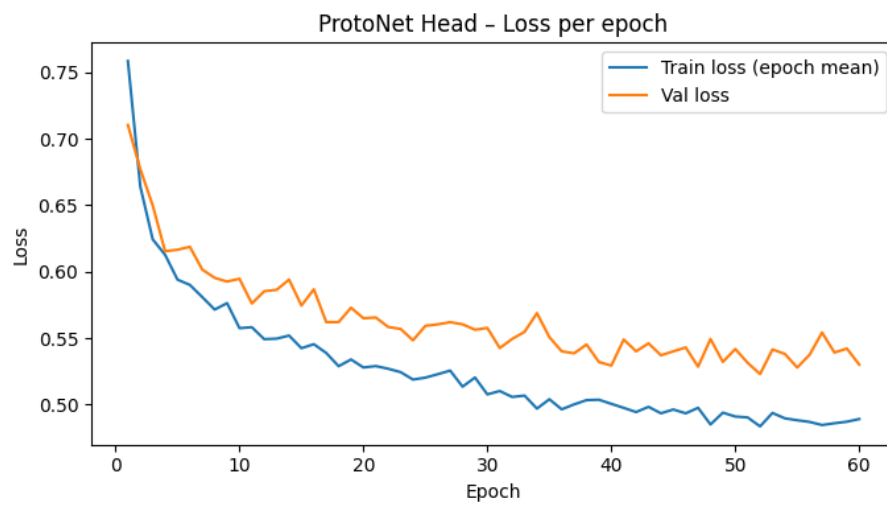


Figura 45: Esperimento 5: andamento della loss (train vs validation) durante l'addestramento.

[Torna alla tabella dei risultati](#)

3.2 Euclidean Distance con ReLU

Add.	n-way	n-shot	Query	Episodi	Epoche	Train Acc	Val Acc	Train F1	Val F1	Gap
1	5	5	15	600	50	≈ 0.800	≈ 0.786	≈ 0.799	≈ 0.784	≈ 1.45%
2	5	3	15	600	50	≈ 0.781	≈ 0.758	≈ 0.778	≈ 0.756	≈ 2.21%
3	5	1	15	600	60	≈ 0.660	≈ 0.628	≈ 0.650	≈ 0.615	≈ 3.17%
4	5	5	15	1000	30	≈ 0.801	≈ 0.790	≈ 0.800	≈ 0.788	≈ 1.13%
5	5	5	15	1000	60	≈ 0.800	≈ 0.796	≈ 0.798	≈ 0.794	≈ 0.41%

3.2.1 Esperimento 1 (n-way=5, n-shot=5, query=15, episodi=600, epoche=50)

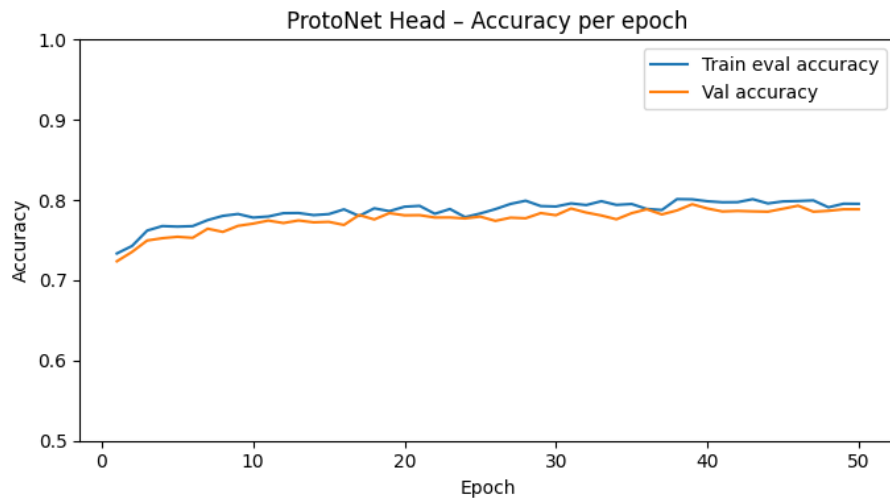


Figura 46: Esperimento 1: andamento dell'accuracy (train vs validation) durante l'addestramento.

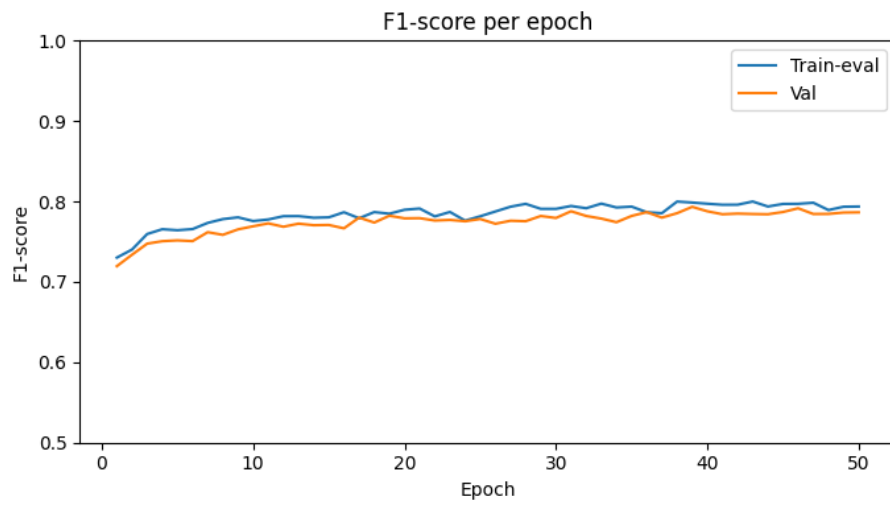


Figura 47: Esperimento 1: andamento del macro F1-score (train vs validation) durante l'addestramento.

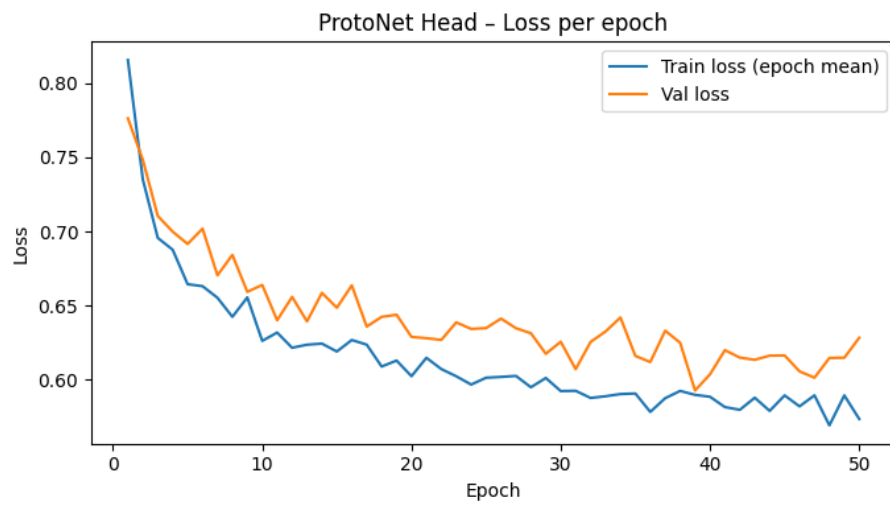


Figura 48: Esperimento 1: andamento della loss (train vs validation) durante l'addestramento.

[Torna alla tabella dei risultati](#)

3.2.2 Esperimento 2 (n-way=5, n-shot=3, query=15, episodi=600, epoche=50)

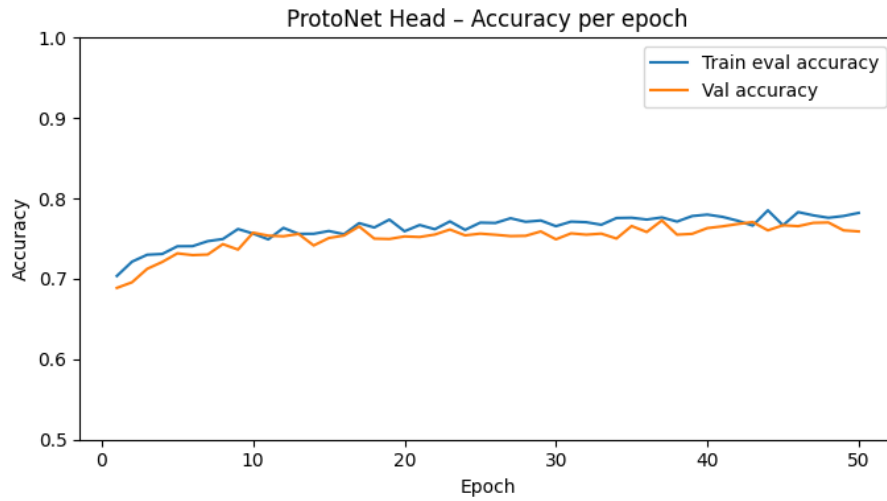


Figura 49: Esperimento 2: andamento dell'accuracy (train vs validation) durante l'addestramento.

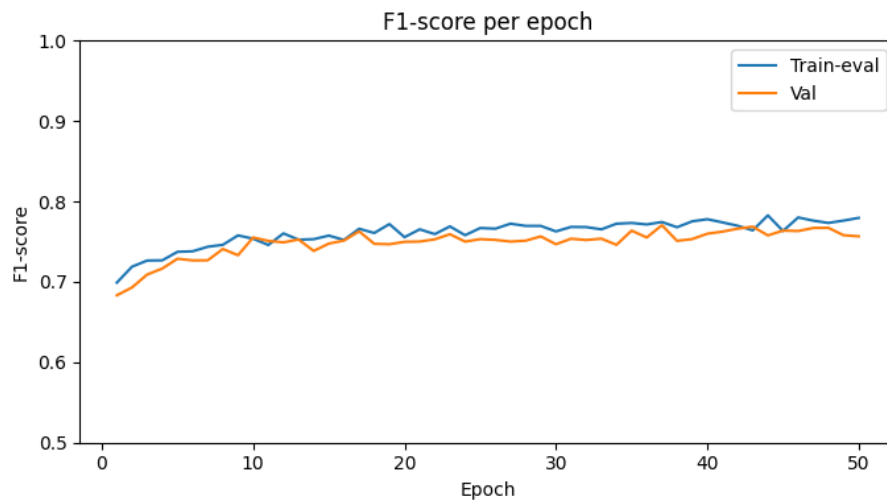


Figura 50: Esperimento 2: andamento del macro F1-score (train vs validation) durante l'addestramento.

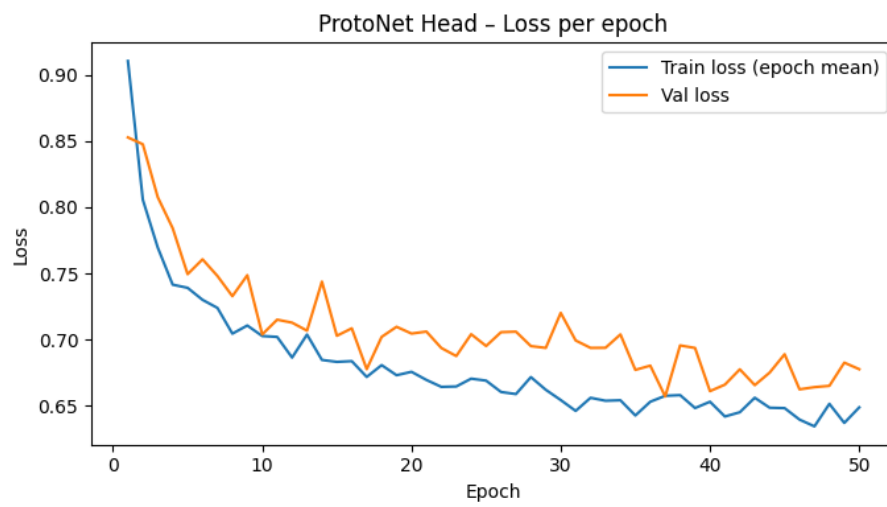


Figura 51: Esperimento 2: andamento della loss (train vs validation) durante l'addestramento.

[Torna alla tabella dei risultati](#)

3.2.3 Esperimento 3 (n-way=5, n-shot=1, query=15, episodi=600, epoche=60)

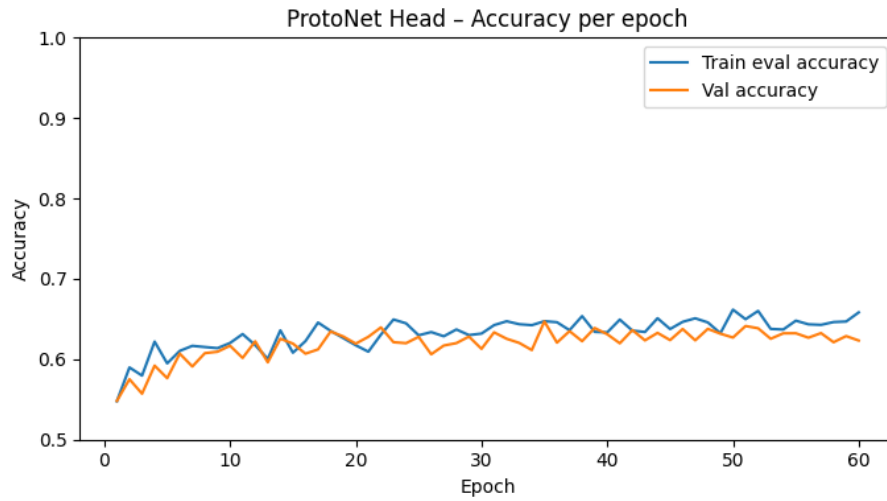


Figura 52: Esperimento 3: andamento dell'accuracy (train vs validation) durante l'addestramento.

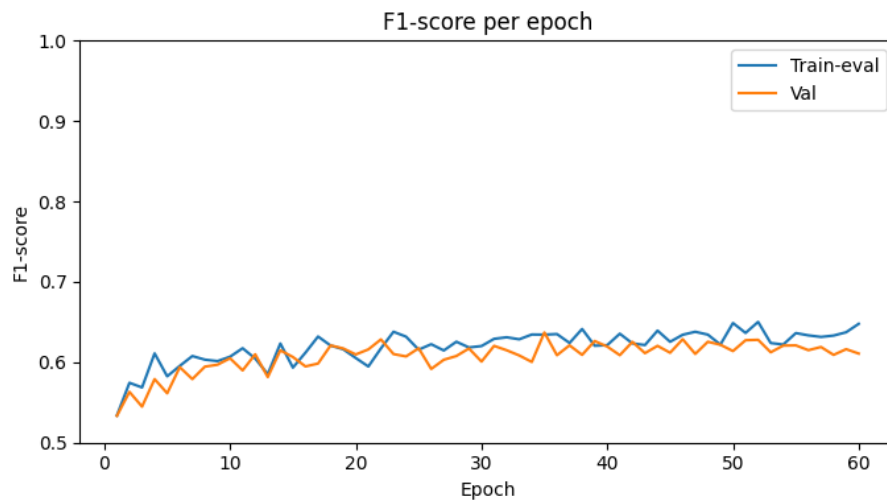


Figura 53: Esperimento 3: andamento del macro F1-score (train vs validation) durante l'addestramento.

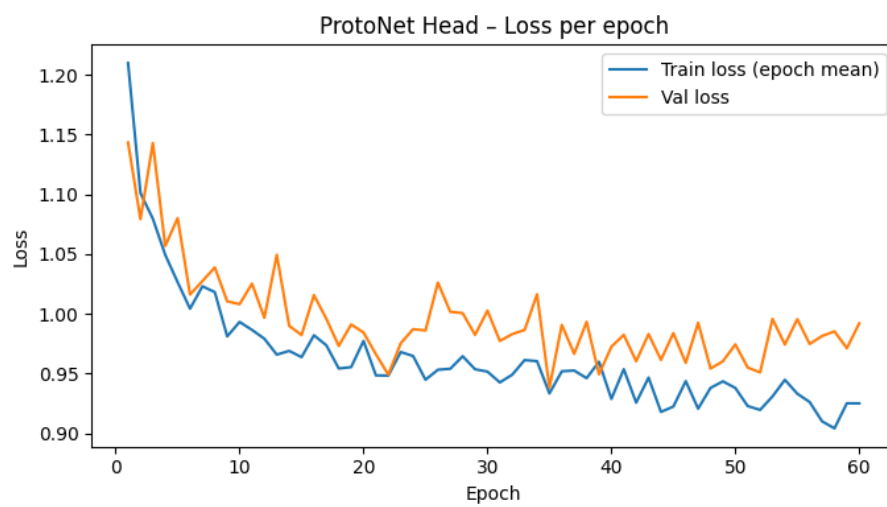


Figura 54: Esperimento 3: andamento della loss (train vs validation) durante l'addestramento.

[Torna alla tabella dei risultati](#)

3.2.4 Esperimento 4 (n-way=5, n-shot=5, query=15, episodi=1000, epoche=30)

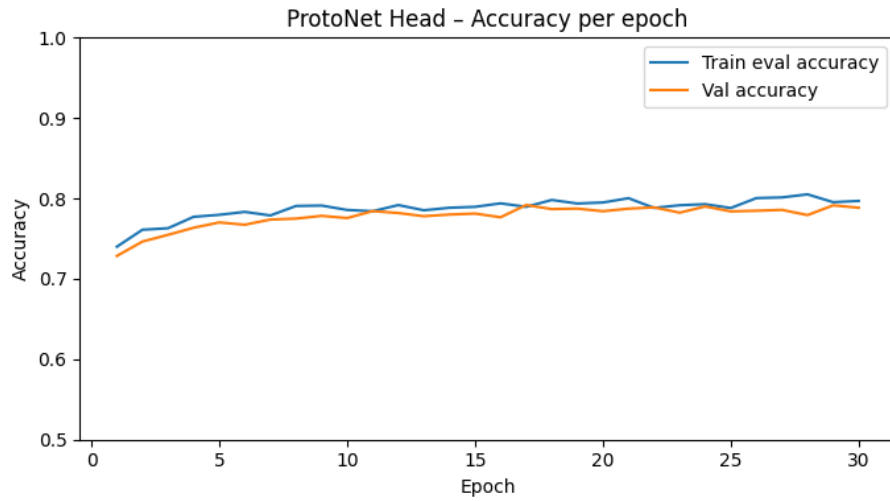


Figura 55: Esperimento 4: andamento dell'accuracy (train vs validation) durante l'addestramento.

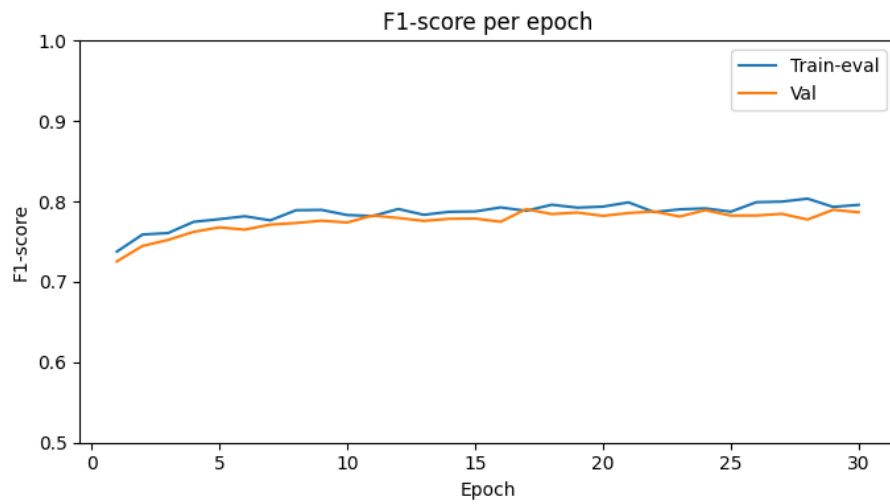


Figura 56: Esperimento 4: andamento del macro F1-score (train vs validation) durante l'addestramento.

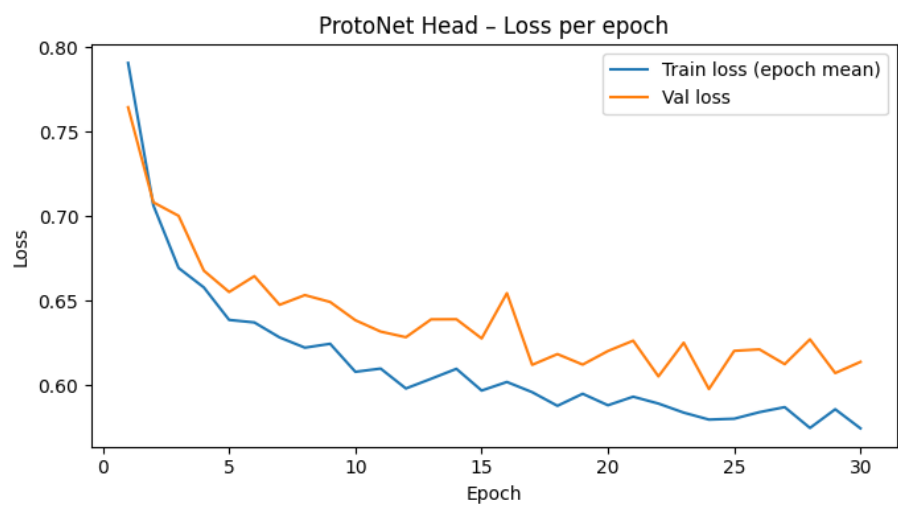


Figura 57: Esperimento 4: andamento della loss (train vs validation) durante l'addestramento.

[Torna alla tabella dei risultati](#)

3.2.5 Esperimento 5 (n-way=5, n-shot=5, query=15, episodi=1000, epoche=60)

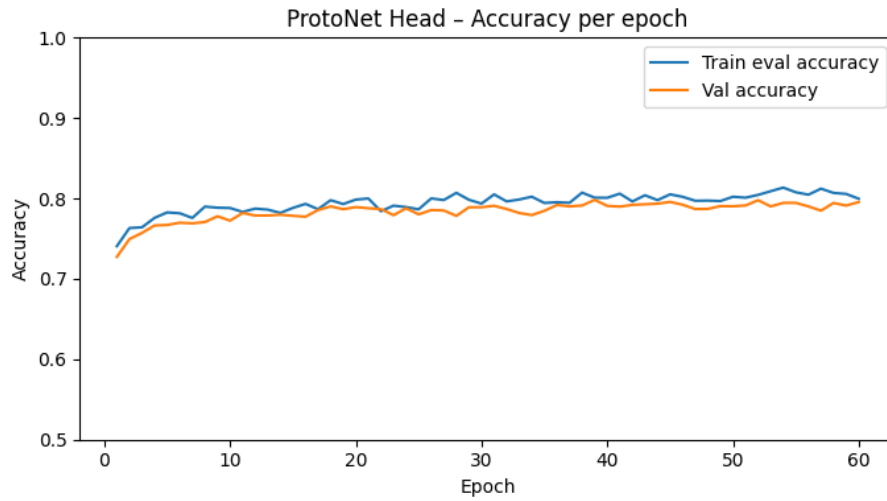


Figura 58: Esperimento 5: andamento dell'accuracy (train vs validation) durante l'addestramento.

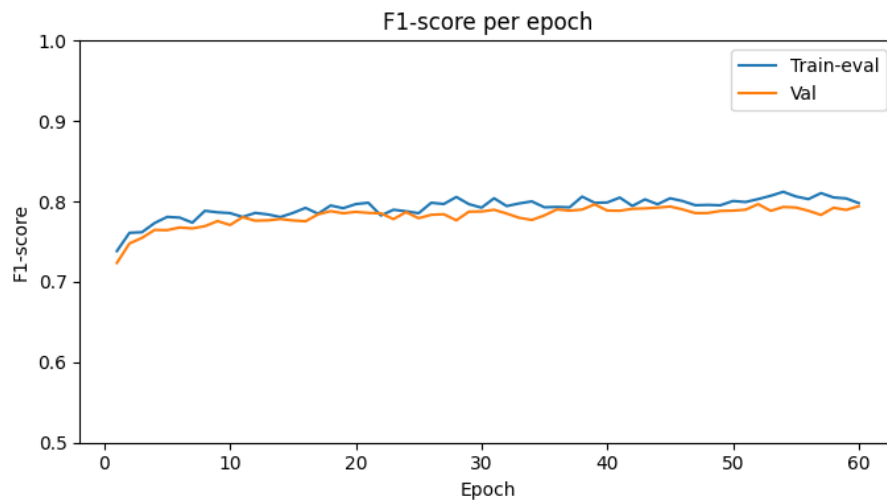


Figura 59: Esperimento 5: andamento del macro F1-score (train vs validation) durante l'addestramento.

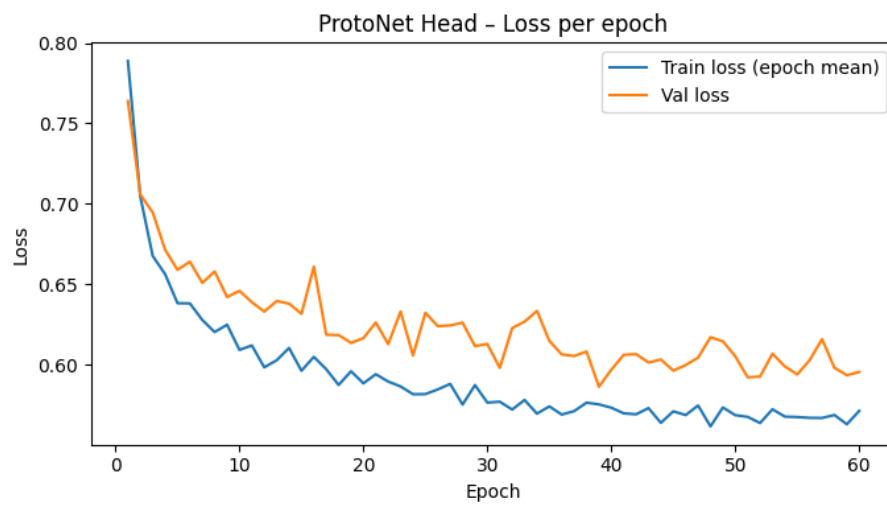


Figura 60: Esperimento 5: andamento della loss (train vs validation) durante l'addestramento.

[Torna alla tabella dei risultati](#)

3.3 Confronto tra i risultati

Exp	n-shot	Episodi	Epoche	Val Acc (Cosine / Euclidean)	Val F1 (Cosine / Euclidean)	Gap migliore
1	5	600	50	0.806 / 0.786	0.805 / 0.784	Euclidean
2	3	600	50	0.785 / 0.758	0.783 / 0.756	Euclidean
3	1	600	60	0.668 / 0.628	0.656 / 0.615	Euclidean
4	5	1000	30	0.805 / 0.790	0.804 / 0.788	Euclidean
5	5	1000	60	0.813 / 0.796	0.812 / 0.794	Euclidean

Tabella 5: Confronto tra distanza cosine e distanza euclidea con head MLP (Linear–ReLU–Linear) per Prototypical Network (solo metriche di validazione). In *Gap migliore* è indicata la metrica con gap train–validation più basso.

Anche nel caso della head MLP (Linear–ReLU–Linear) si osserva lo stesso scenario emerso con la head puramente lineare: la cosine similarity ottiene sistematicamente valori leggermente migliori in termini di accuratezza e macro F1-score di validazione, indicando una maggiore efficacia nella separazione delle classi all’interno degli episodi few-shot.

Nonostante le prestazioni assolute più elevate, la cosine similarity presenta un gap train–validation generalmente più ampio. Al contrario, la distanza euclidea, pur penalizzando marginalmente accuracy e F1-score, mostra anche in questo caso un gap più contenuto, suggerendo una migliore capacità di generalizzazione.

Questo comportamento può essere ricondotto al fatto che, in presenza di una head non lineare, la distanza euclidea impone un vincolo geometrico più restrittivo sullo spazio delle rappresentazioni, limitando l’adattamento eccessivo ai dati di training episodico. Di conseguenza, la metrica euclidea risulta più robusta rispetto alle variazioni tra training e validation, confermando il trend già osservato negli esperimenti precedenti.

4 Cosine similarity con Tanh

Un componente critico nell'architettura delle Prototypical Networks è la funzione di attivazione non lineare che trasforma gli embedding prima del calcolo della distanza. In questo studio, abbiamo confrontato l'approccio standard basato su ReLU e una variante basata su Tanh, motivata dalla natura geometrica dello spazio metrico utilizzato. La ReLU (Rectified Linear Unit), matematicamente definita come $f(x) = \max(0, x)$, azzerava tutti i valori negativi lasciando inalterati quelli positivi. Sebbene sia standard nel Deep Learning, il suo utilizzo in combinazione con la Cosine Similarity presenta una limitazione teorica: azzerando le componenti negative, la ReLU confina i vettori delle feature esclusivamente nel 'quadrante positivo' dello spazio vettoriale. Questo riduce drasticamente lo spazio espressivo disponibile, rendendo impossibile per il modello rappresentare feature 'opposte' o negativamente correlate. Al contrario, la Tanh (Tangente Iperbolica) è una funzione sigmoidale centrata sullo zero che mappa qualsiasi input nell'intervallo $[-1, 1]$. Il suo obiettivo è preservare il segno dell'informazione: i valori fortemente negativi tendono a -1, quelli positivi a +1.

Abbiamo ipotizzato che l'adozione della Tanh sia superiore in questo contesto per due motivi principali:

Coerenza Geometrica: Mappando i valori in $[-1, 1]$, la Tanh permette ai vettori di sfruttare l'intera ipersfera direzionale. La 'dissimilarità' viene codificata non solo come ortogonalità (0), ma anche come opposizione (-1), arricchendo la capacità discriminativa.

Centratura dei Dati: Gli embedding estratti da backbone pre-addestrati (come DINOv3) contengono spesso informazioni significative distribuite attorno allo zero; la Tanh rispetta questa distribuzione evitando la perdita di informazione causata dal troncamento della ReLU.

Add.	n-way	n-shot	Query	Episodi	Epoche	Train Acc	Val Acc	Train F1	Val F1	Gap
1	5	5	15	600	50	0.8196	0.8039	0.8179	0.8019	1.57%
2	5	3	15	600	50	0.8108	0.7823	0.8092	0.7809	2.85%
3	5	1	15	600	60	0.7042	0.6727	0.6937	0.6601	3.15%
4	5	5	15	1000	30	0.8161	0.8065	0.8150	0.8054	0.95%
5	5	5	15	1000	60	0.8231	0.8142	0.8225	0.8132	0.89%

Tabella 6: Configurazione di addestramento del modello Prototypical Network con attivazione *tanh* (distanza cosine, learning rate 1×10^{-3}).

4.1 Esperimento 1 (n-way=5, n-shot=5, query=15, episodi=600, epoche=50)

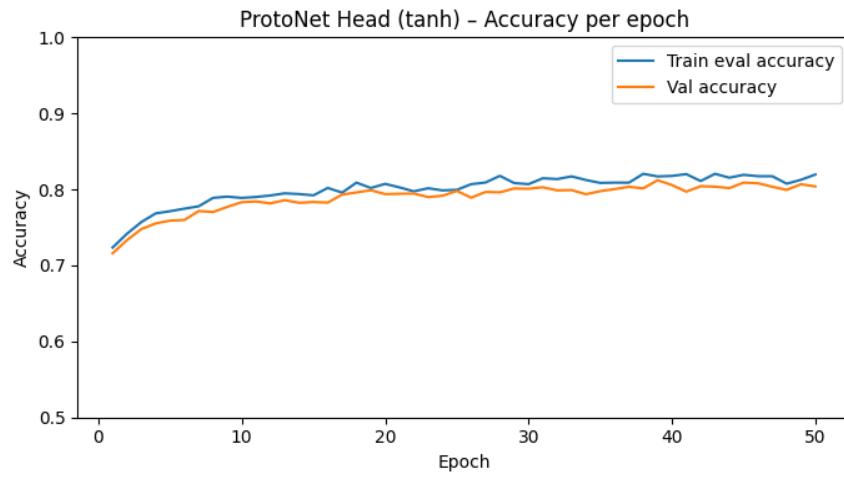


Figura 61: Accuracy - Esperimento 1 (Cosine + Tanh).

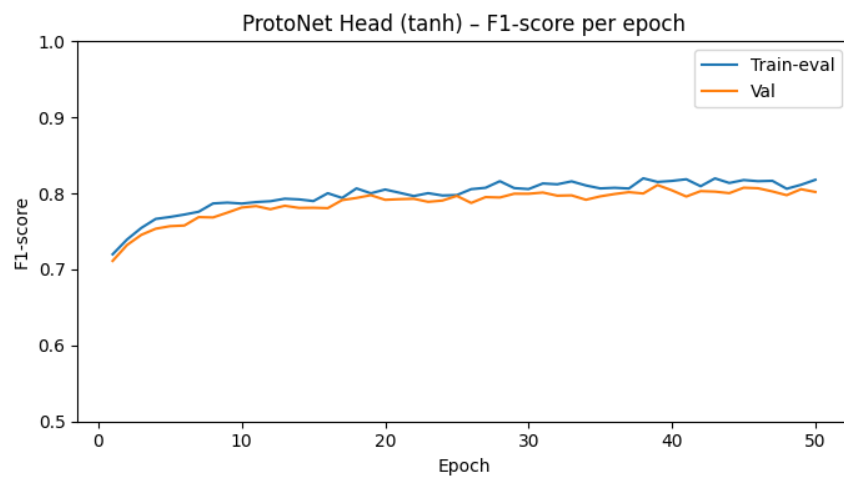


Figura 62: F1-score - Esperimento 1 (Cosine + Tanh).

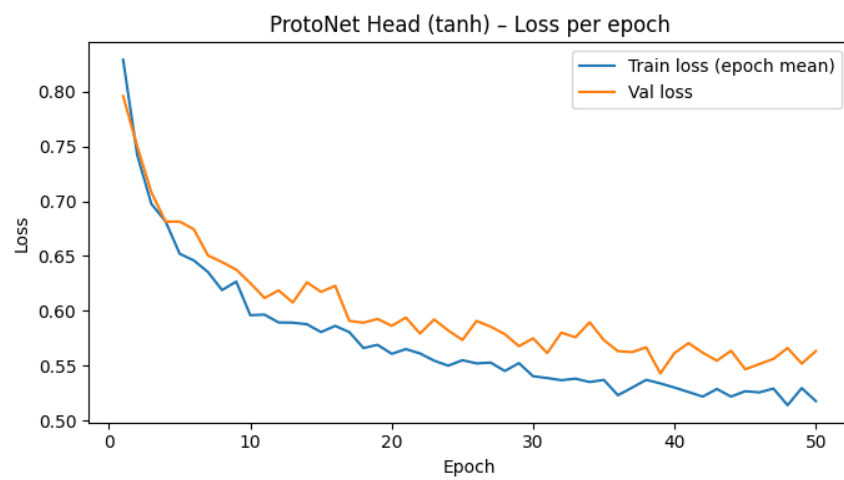


Figura 63: Loss - Esperimento 1 (Cosine + Tanh).

4.2 Esperimento 2 (n-way=5, n-shot=3, query=15, episodi=600, epoche=50)

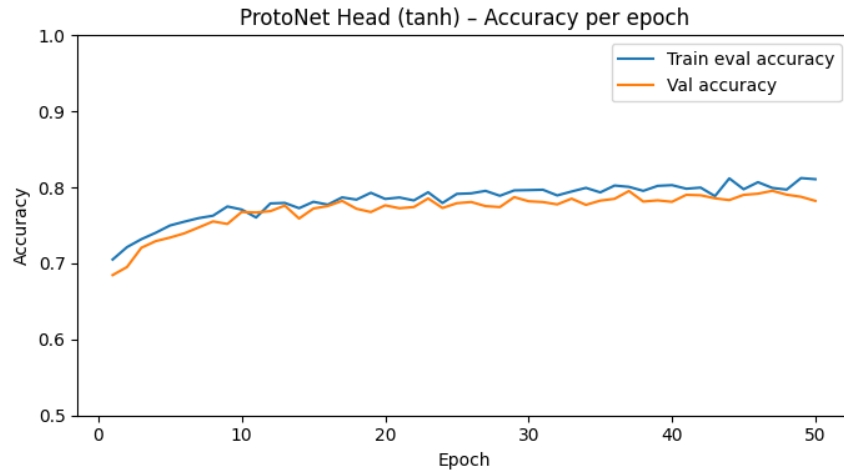


Figura 64: Accuracy - Esperimento 2 (Cosine + Tanh).

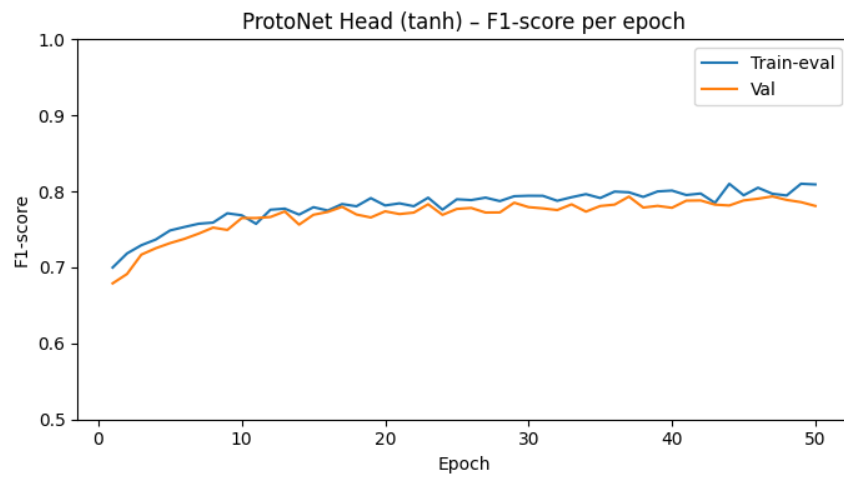


Figura 65: F1-score - Esperimento 2 (Cosine + Tanh).

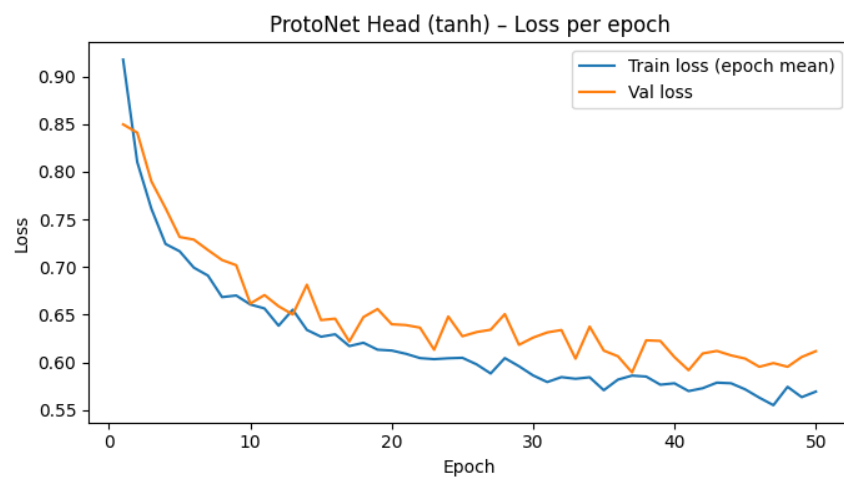


Figura 66: Loss - Esperimento 2 (Cosine + Tanh).

4.3 Esperimento 3 (n-way=5, n-shot=1, query=15, episodi=600, epoche=60)

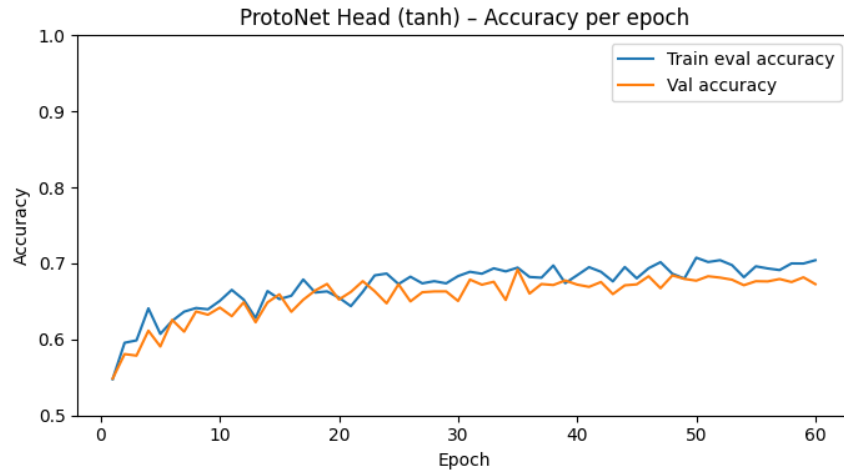


Figura 67: Accuracy - Esperimento 3 (Cosine + Tanh).

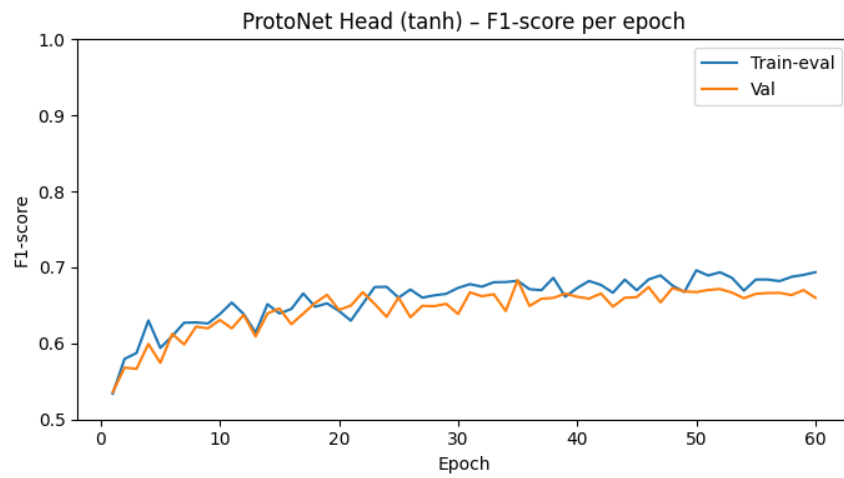


Figura 68: F1-score - Esperimento 3 (Cosine + Tanh).

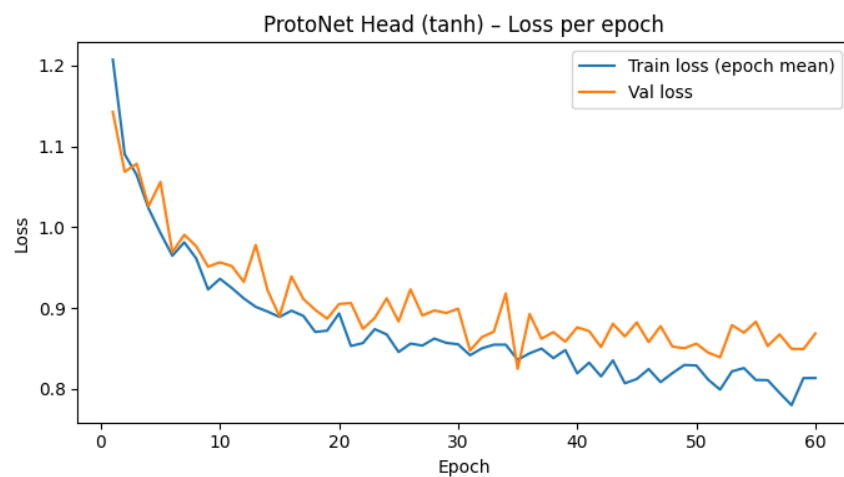


Figura 69: Loss - Esperimento 3 (Cosine + Tanh).

4.4 Esperimento 4 (n-way=5, n-shot=5, query=15, episodi=1000, epoche=30)

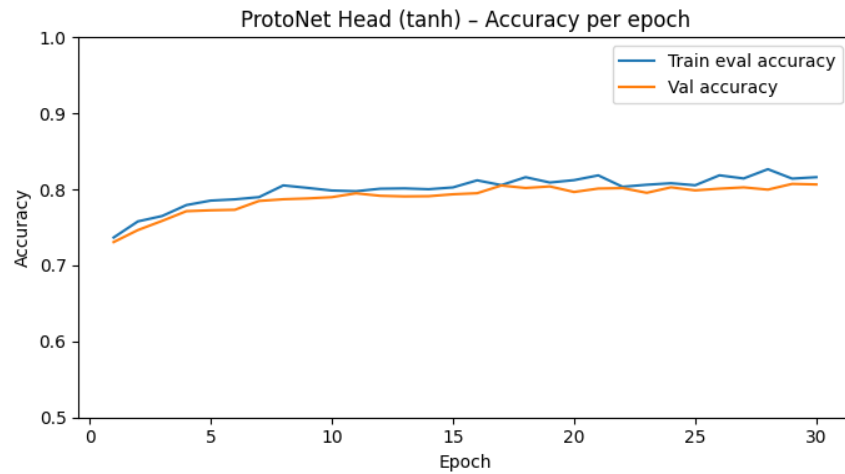


Figura 70: Accuracy - Esperimento 4 (Cosine + Tanh).

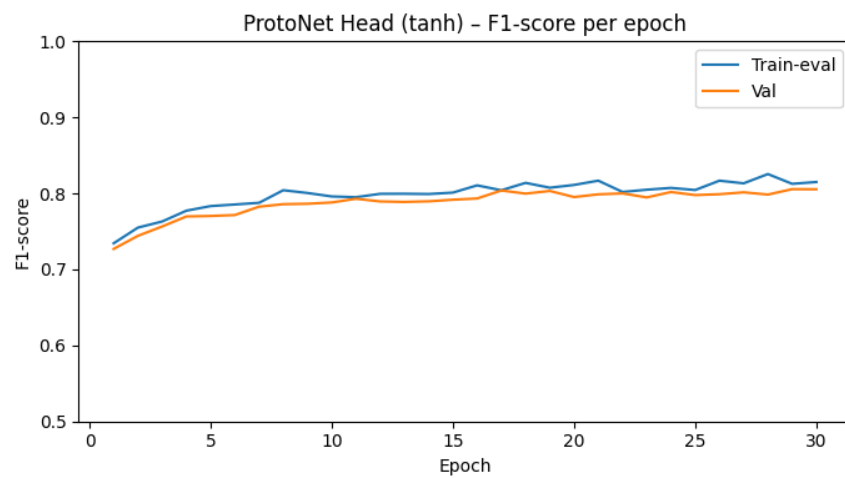


Figura 71: F1-score - Esperimento 4 (Cosine + Tanh).

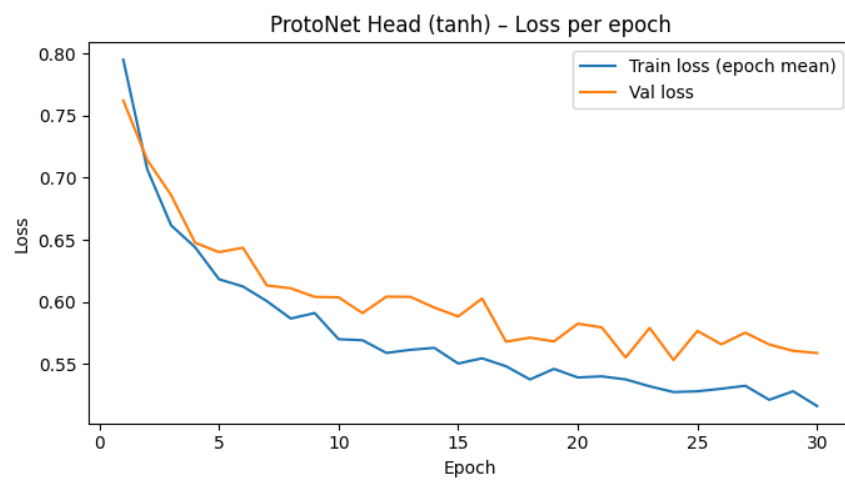


Figura 72: Loss - Esperimento 4 (Cosine + Tanh).

4.5 Esperimento 5 (n-way=5, n-shot=5, query=15, episodi=1000, epoche=60)

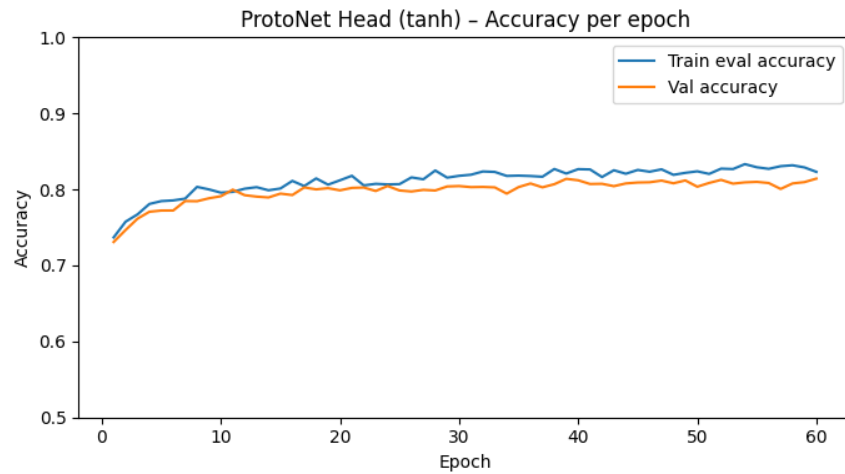


Figura 73: Accuracy - Esperimento 5 (Cosine + Tanh).

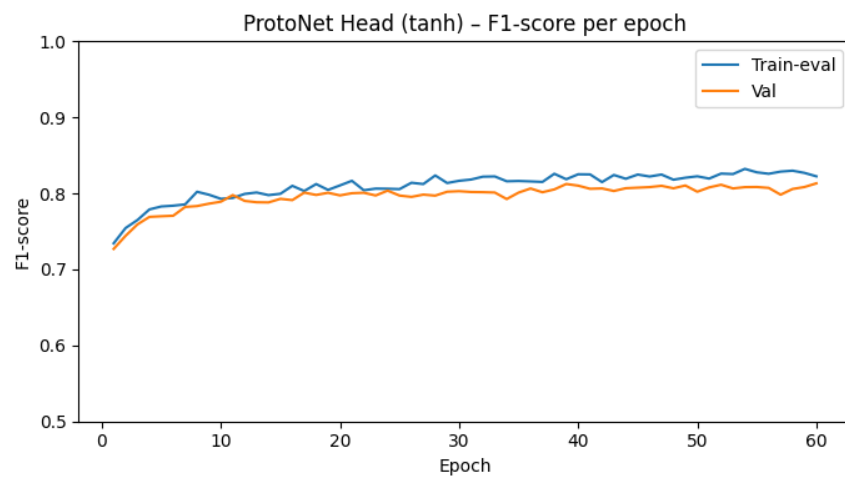


Figura 74: F1-score - Esperimento 5 (Cosine + Tanh).

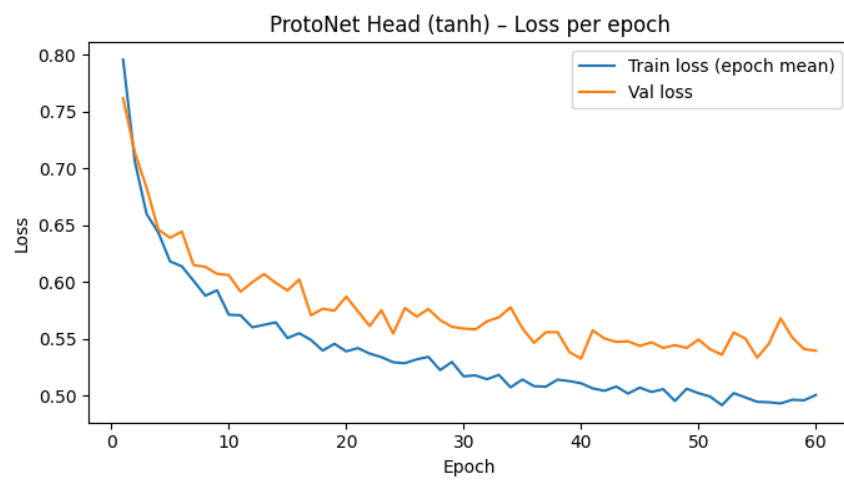


Figura 75: Loss - Esperimento 5 (Cosine + Tanh).

[Torna alla tabella dei risultati](#)

5 Confronto risultati cosine tra Relu e Tanh

Dall'analisi dei risultati sperimentali emerge che ReLU è leggermente migliore in media negli scenari meno critici quali gli esperimenti 1 e 2 mentre tanh diventa vantaggiosa negli scenari più complessi o lunghi. Questo comportamento è coerente con le proprietà geometriche della funzione *tanh* che introduce una normalizzazione implicita delle attivazioni e favorisce una rappresentazione più compatta nello spazio degli embedding.

Exp	n-shot	Episodi	Epoche	Train Acc (tanh)	Train Acc (ReLU)	Val Acc (tanh)	Val Acc (ReLU)	Gap migliore
1	5	600	50	0.8196	0.8220	0.8039	0.8060	ReLU
2	3	600	50	0.8108	0.8110	0.7823	0.7850	ReLU
3	1	600	60	0.7042	0.7040	0.6727	0.6680	tanh
4	5	1000	30	0.8161	0.8180	0.8065	0.8050	tanh
5	5	1000	60	0.8231	0.8260	0.8142	0.8130	tanh

Tabella 7: Confronto tra Cosine Similarity con attivazione *tanh* e con ReLU (accuratezza su train e validation).

5.1 Esperimento 1

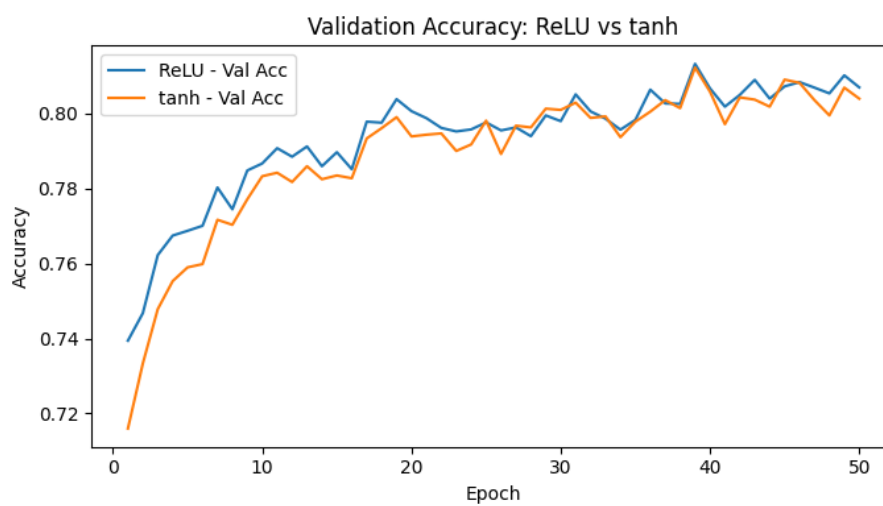


Figura 76: Esperimento 1: andamento dell'accuracy (validation).

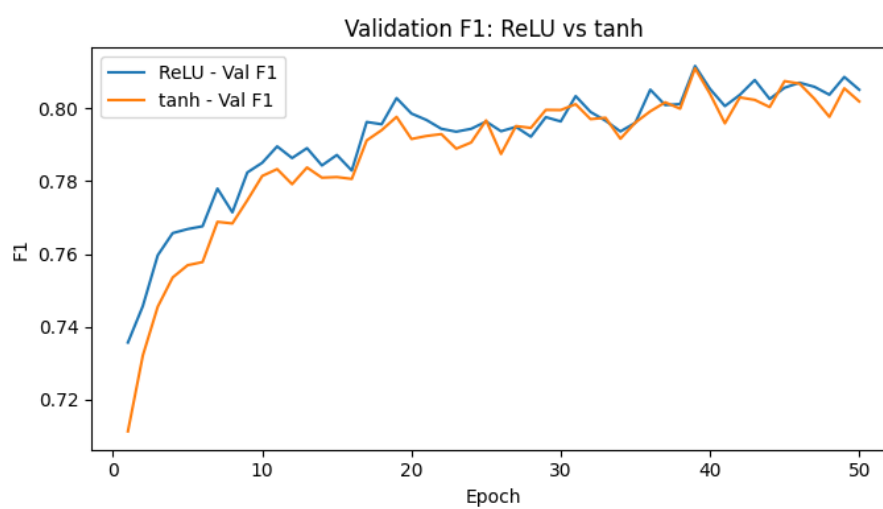


Figura 77: Esperimento 1: andamento del macro F1-score (validation).

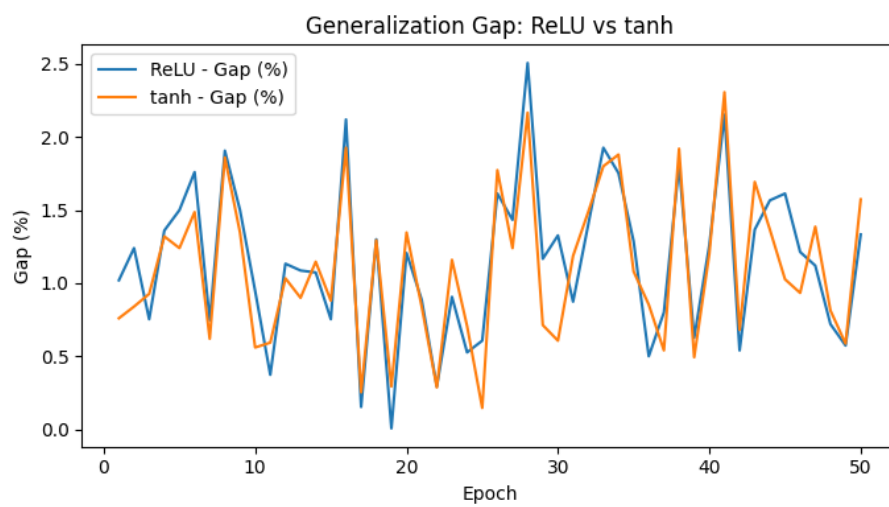


Figura 78: Esperimento 1: andamento del gap train-validation (validation).

[Torna alla tabella dei risultati](#)

5.2 Esperimento 2

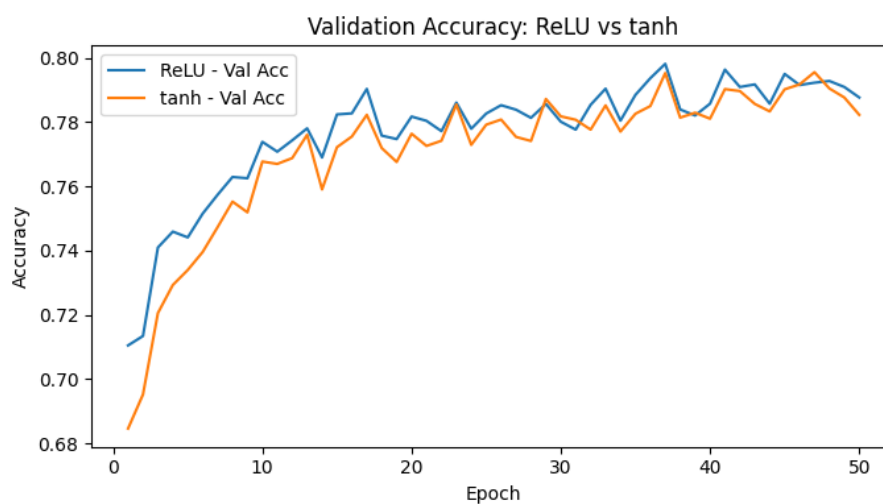


Figura 79: Esperimento 2: andamento dell'accuracy (train vs validation).

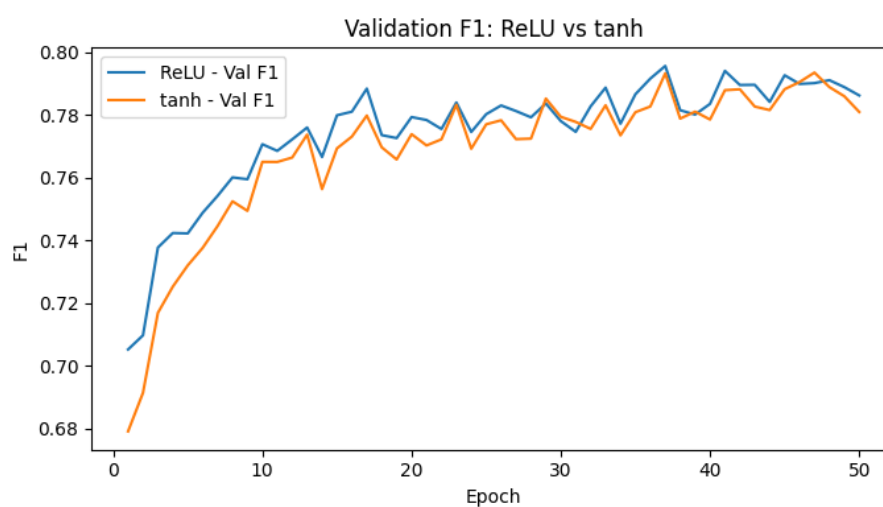


Figura 80: Esperimento 2: andamento del macro F1-score (train vs validation).

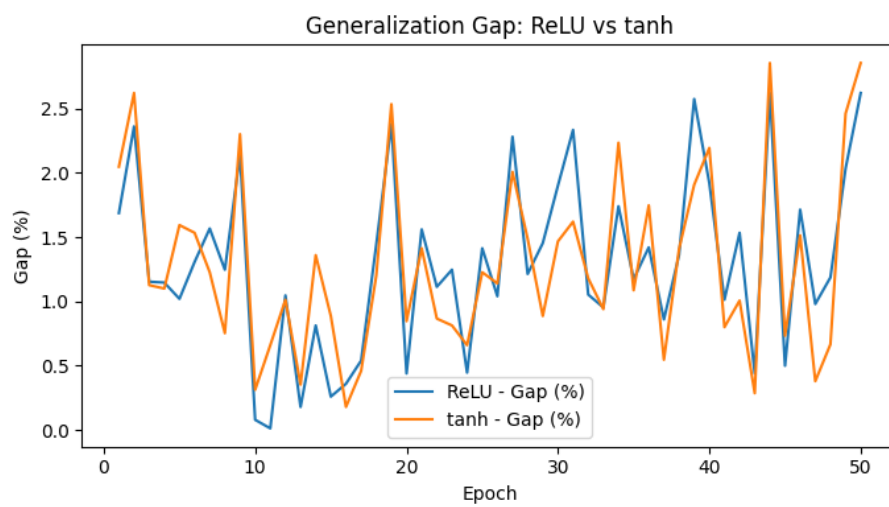


Figura 81: Esperimento 2: andamento del gap train-validation.

[Torna alla tabella dei risultati](#)

5.3 Esperimento 3

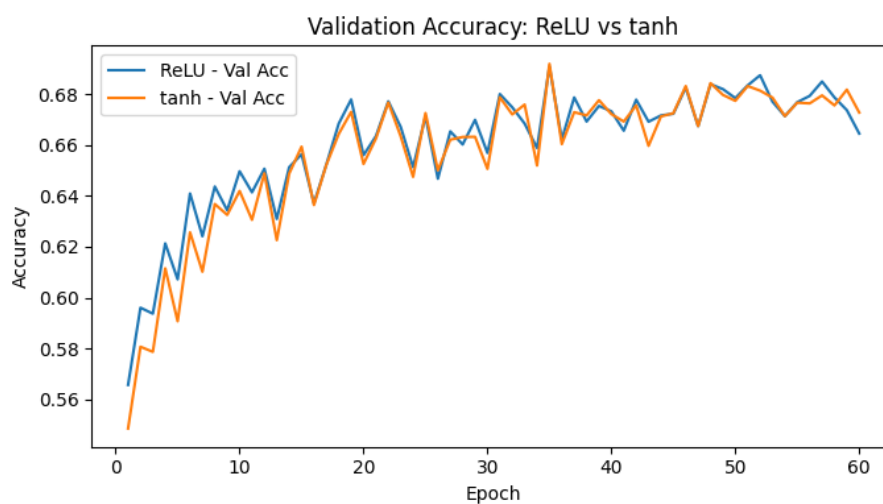


Figura 82: Esperimento 3: andamento dell'accuracy (train vs validation).

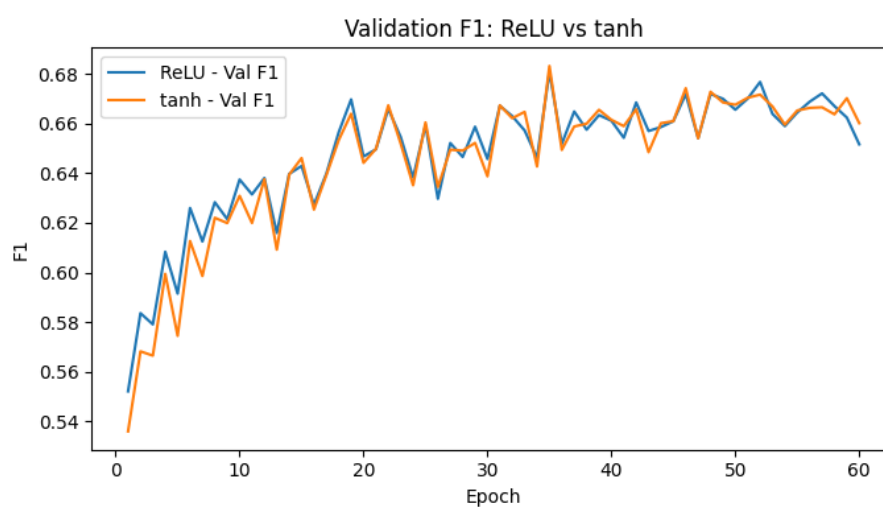


Figura 83: Esperimento 3: andamento del macro F1-score (train vs validation).

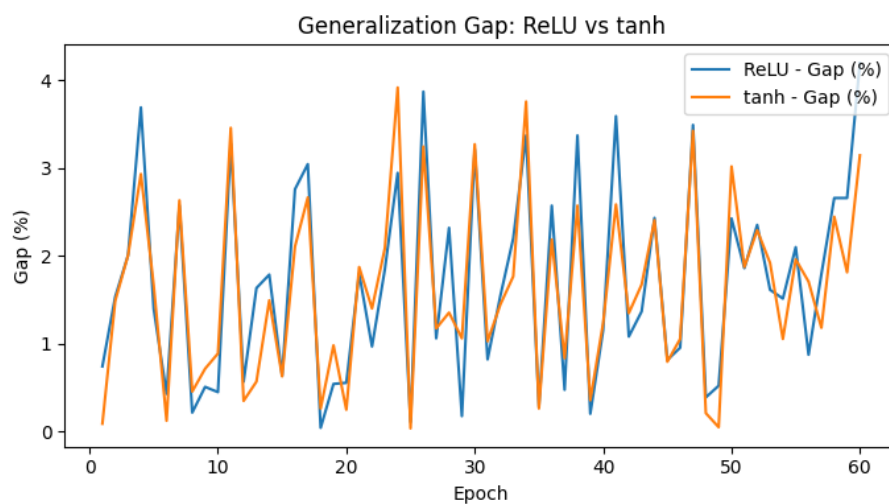


Figura 84: Esperimento 3: andamento del gap train-validation.

[Torna alla tabella dei risultati](#)

5.4 Esperimento 4

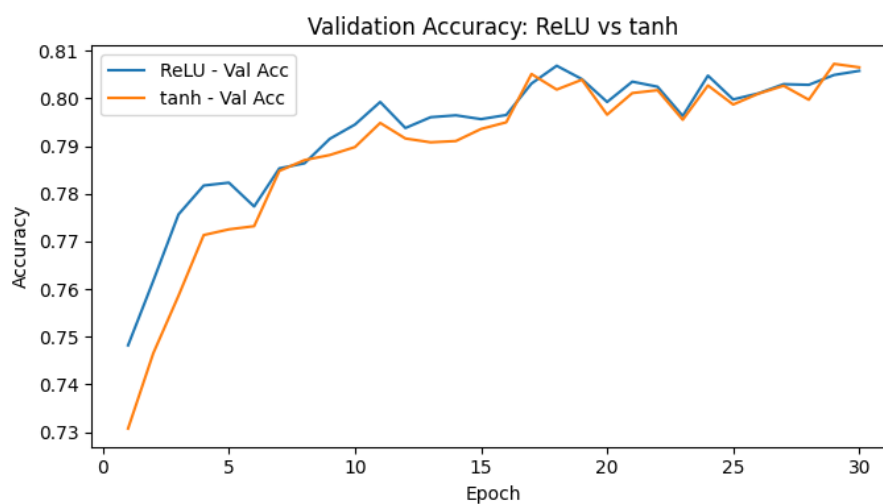


Figura 85: Esperimento 4: andamento dell'accuracy (train vs validation).

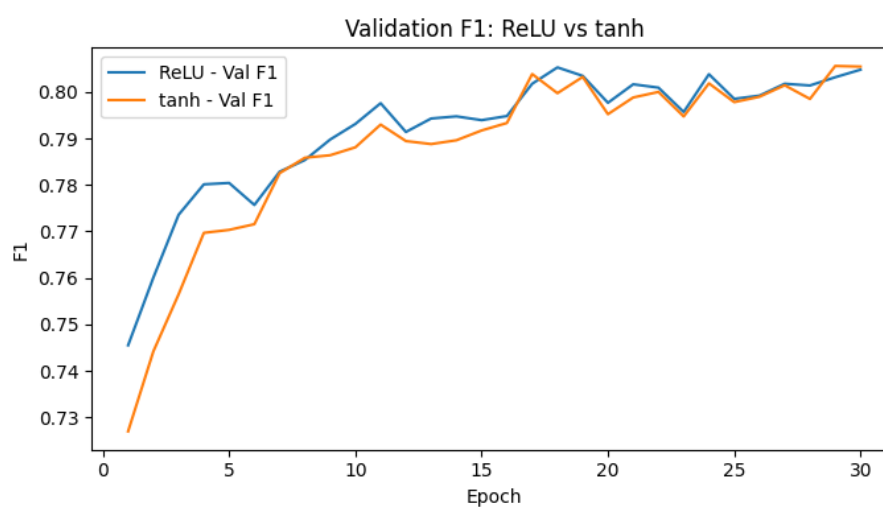


Figura 86: Esperimento 4: andamento del macro F1-score (train vs validation).

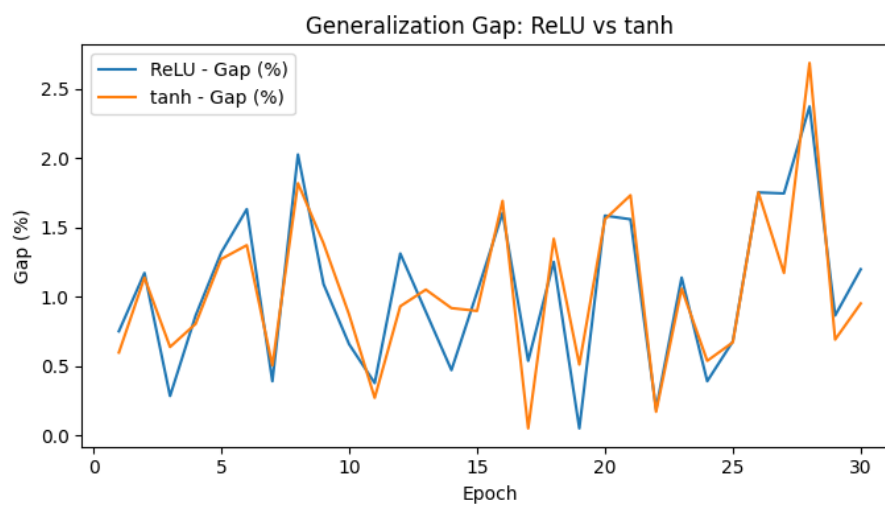


Figura 87: Esperimento 4: andamento del gap train-validation.

[Torna alla tabella dei risultati](#)

5.5 Esperimento 5

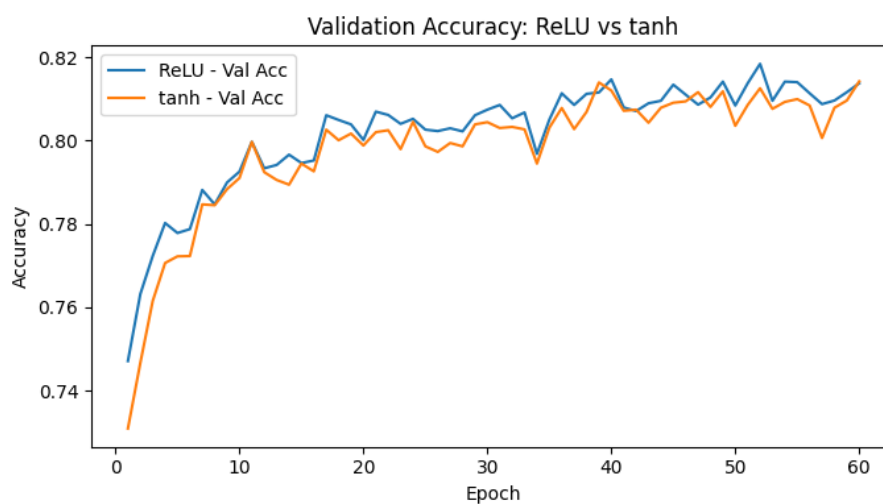


Figura 88: Esperimento 5: andamento dell'accuracy (train vs validation).

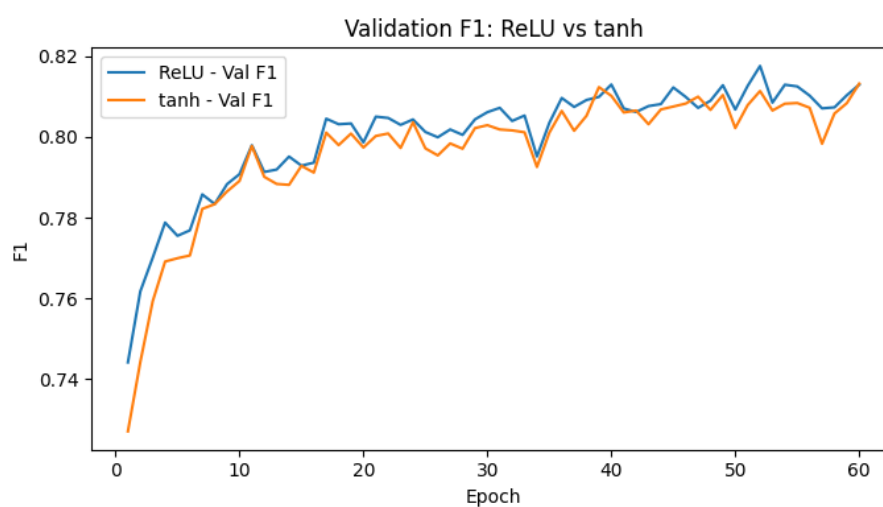


Figura 89: Esperimento 5: andamento del macro F1-score (train vs validation).

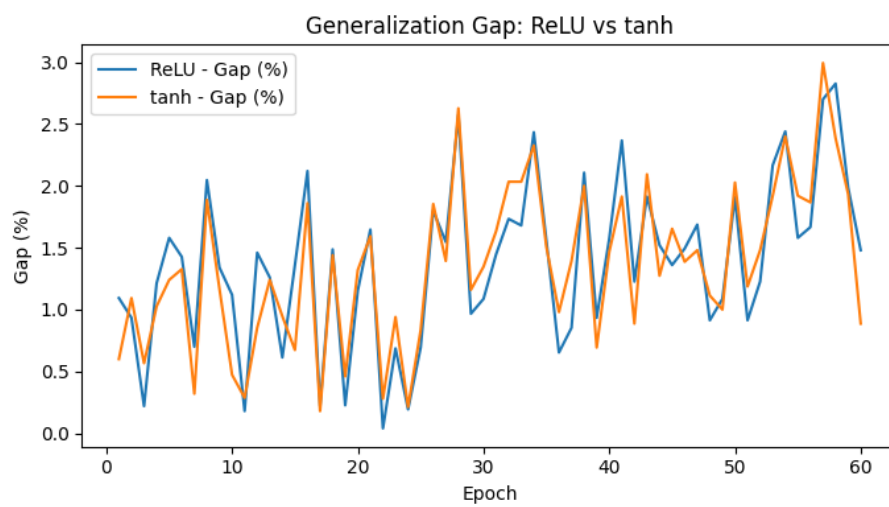


Figura 90: Esperimento 5: andamento del gap train-validation.

[Torna alla tabella dei risultati](#)

6 Ottimizzazione con SGD e Learning Rate Scheduler

In questa fase del lavoro è stata analizzata l'influenza dell'algoritmo di ottimizzazione sulle prestazioni del classificatore, con particolare attenzione allo Stochastic Gradient Descent. Sebbene ottimizzatori adattivi come AdamW siano spesso preferiti per la loro rapidità di convergenza, SGD rappresenta una scelta di interesse per valutare se un processo di apprendimento più controllato possa favorire una migliore generalizzazione, soprattutto in scenari Few-Shot.

Il learning rate determina l'ampiezza degli aggiornamenti applicati ai pesi del modello in risposta all'errore di predizione. Nel caso di SGD, tale parametro è globale e uniforme per tutti i pesi, rendendo l'ottimizzazione fortemente dipendente dalla sua corretta impostazione. Questa caratteristica, se da un lato rende l'addestramento più sensibile alla scelta degli iperparametri, dall'altro consente un controllo più diretto sulla dinamica di apprendimento, riducendo il rischio di aggiornamenti troppo aggressivi su parametri instabili.

Per mitigare le limitazioni di un learning rate fisso, è stato introdotto un learning rate scheduler, che consente di ridurre progressivamente il tasso di apprendimento durante l'addestramento. Tale strategia permette di combinare una fase iniziale di apprendimento più esplorativa con una fase finale di affinamento dei pesi, favorendo una convergenza più stabile e potenzialmente una migliore capacità di generalizzazione.

L'analisi dell'ottimizzazione con SGD, in confronto agli approcci adattivi, consente quindi di valutare se una dinamica di apprendimento più semplice ma meglio regolata possa risultare vantaggiosa nel contesto considerato.

6.1 ReLU

Add.	n-way	n-shot	Query	Episodi	Epoche	lr	weight decay	Train Acc	Val Acc	Train F1	Val F1	Gap
1	5	5	15	600	100	10^{-2}	10^{-4}	≈ 0.823	≈ 0.810	≈ 0.821	≈ 0.809	$\approx 1.28\%$
2	5	5	15	600	100	5×10^{-3}	10^{-4}	≈ 0.818	≈ 0.807	≈ 0.817	≈ 0.805	$\approx 1.16\%$
3	5	5	15	1000	100	10^{-2}	0	≈ 0.830	≈ 0.812	≈ 0.829	≈ 0.810	$\approx 1.85\%$

6.2 Esperimento 1 (ReLU)

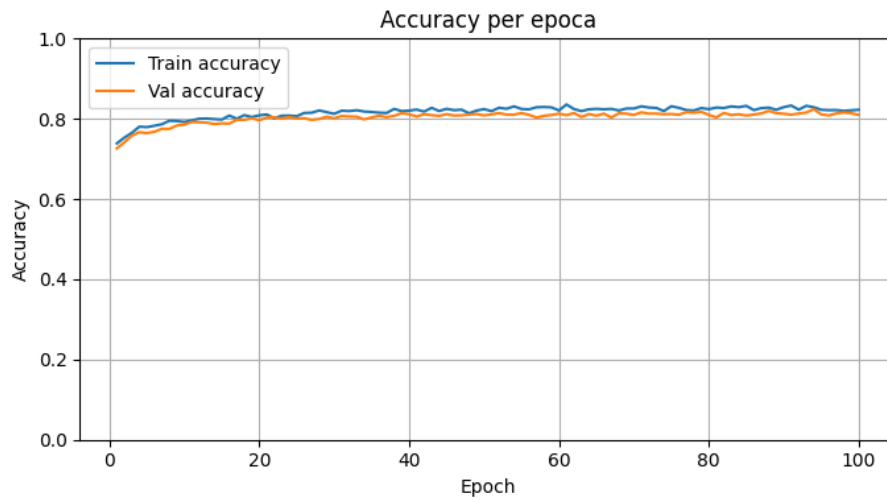


Figura 91: Esperimento 1 (ReLU): andamento dell'accuracy (train vs validation).

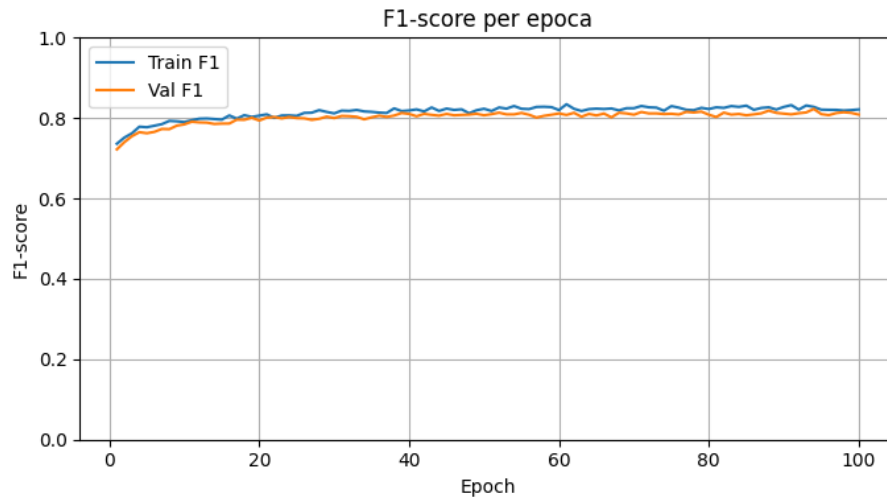


Figura 92: Esperimento 1 (ReLU): andamento del macro F1-score (train vs validation).

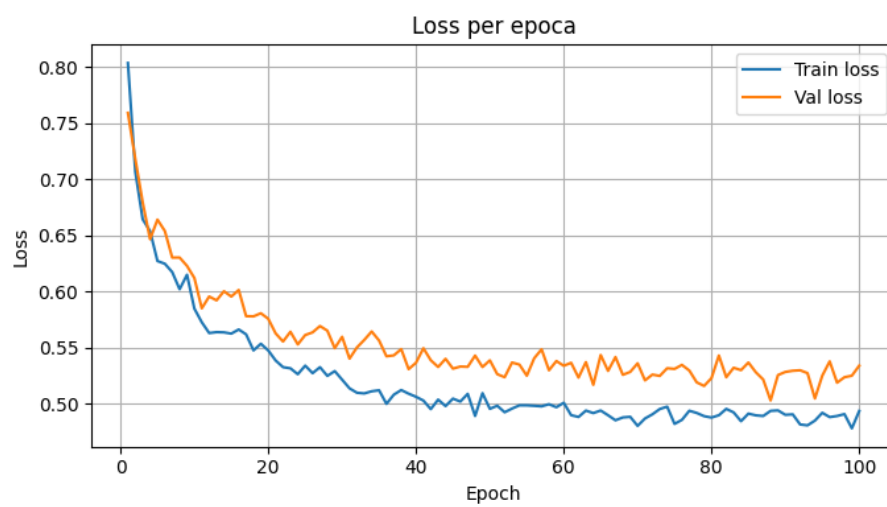


Figura 93: Esperimento 1 (ReLU): andamento della loss (train vs validation).

[Torna alla tabella dei risultati](#)

6.3 Esperimento 2 (ReLU)

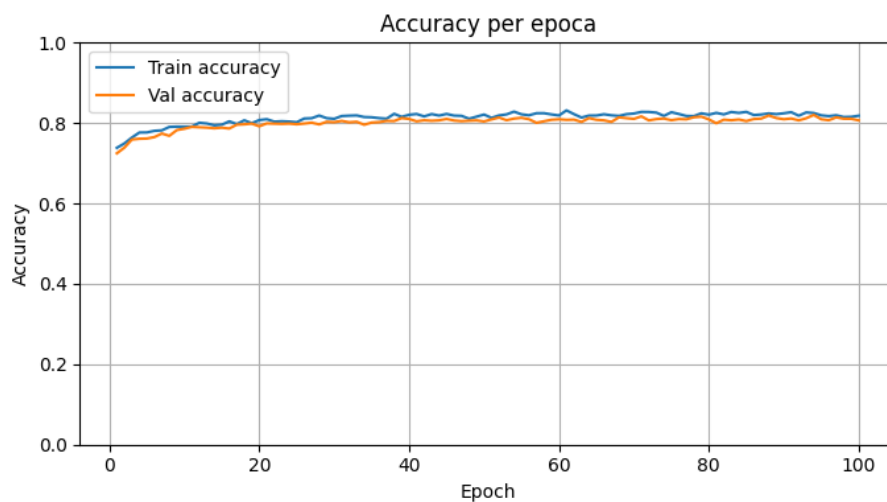


Figura 94: Esperimento 2 (ReLU): andamento dell'accuracy (train vs validation).

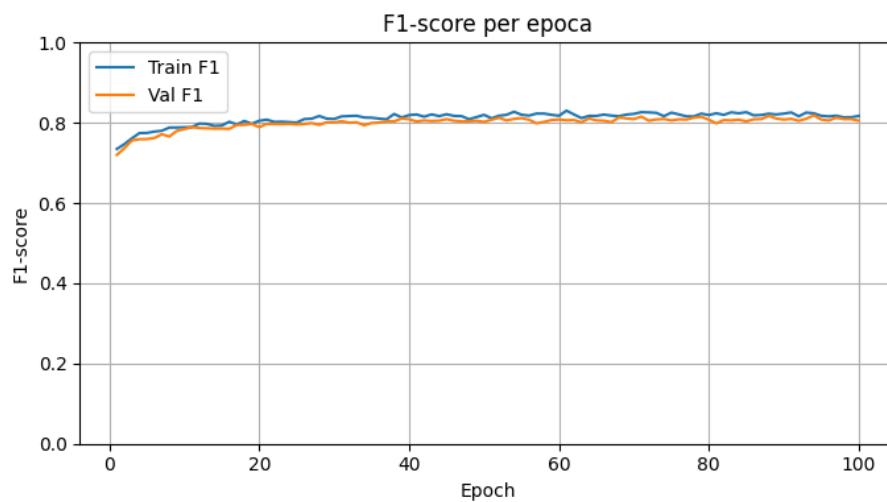


Figura 95: Esperimento 2 (ReLU): andamento del macro F1-score (train vs validation).

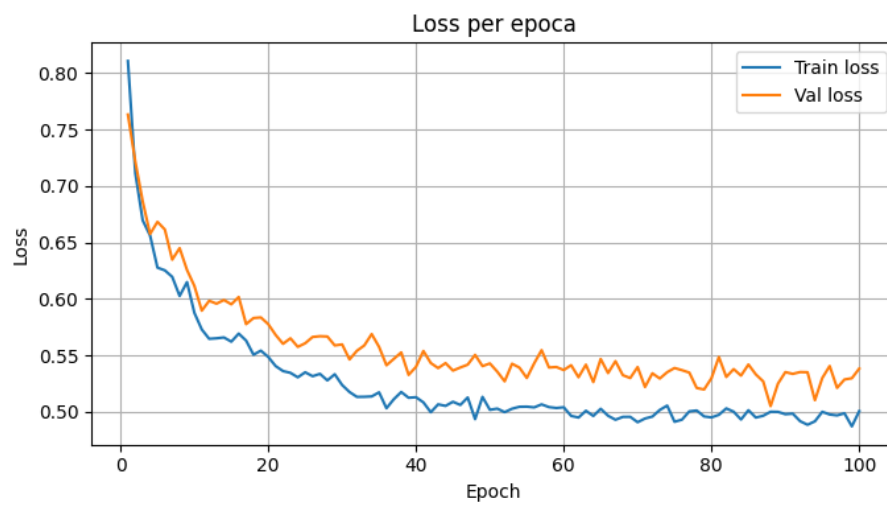


Figura 96: Esperimento 2 (ReLU): andamento della loss (train vs validation).

[Torna alla tabella dei risultati](#)

6.4 Esperimento 3 (ReLU)

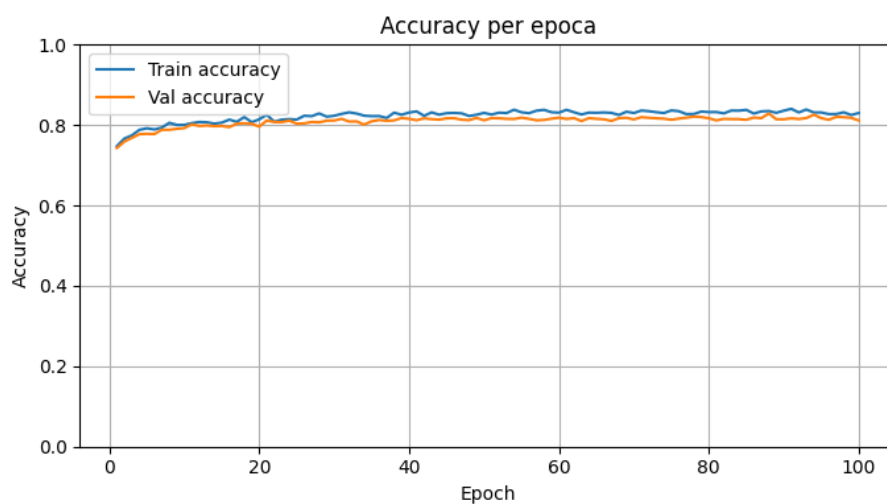


Figura 97: Esperimento 3 (ReLU): andamento dell'accuracy (train vs validation).

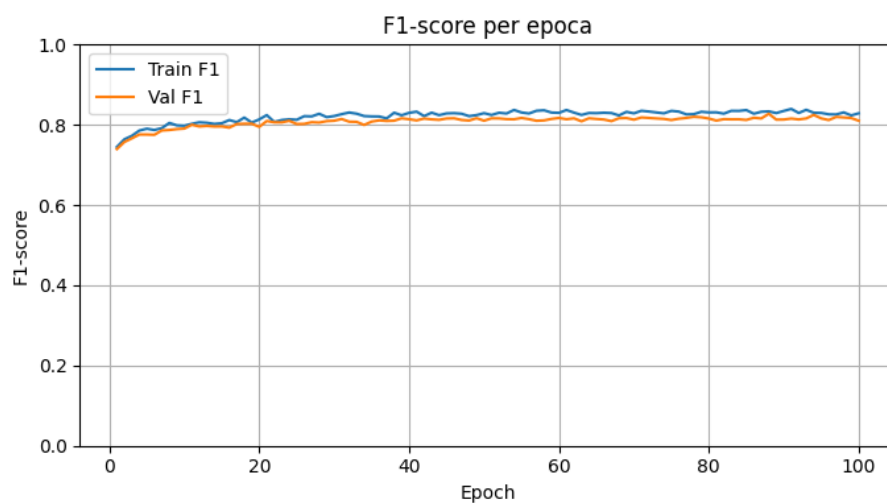


Figura 98: Esperimento 3 (ReLU): andamento del macro F1-score (train vs validation).

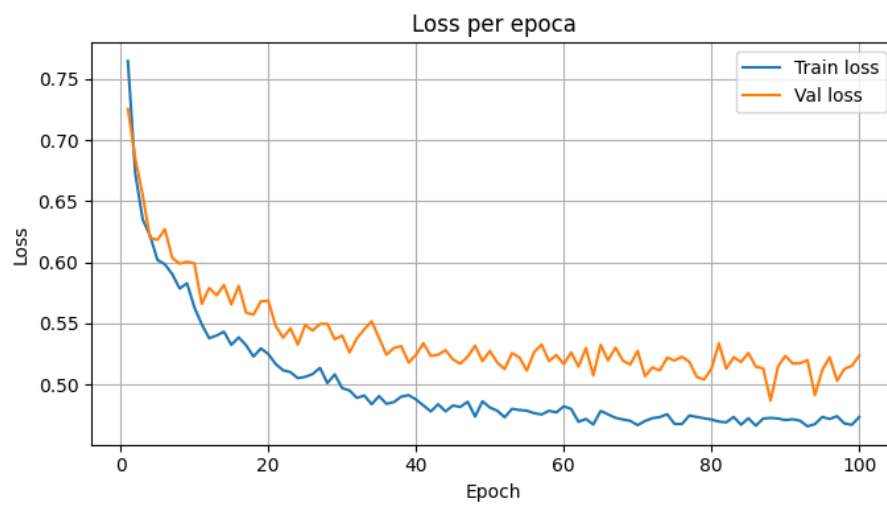


Figura 99: Esperimento 3 (ReLU): andamento della loss (train vs validation).

[Torna alla tabella dei risultati](#)

6.5 Tanh

Add.	n-way	n-shot	Query	Episodi	Epoche	lr	weight decay	Train Acc	Val Acc	Train F1	Val F1	Gap
1	5	5	15	600	100	10^{-2}	10^{-4}	0.8092	0.7964	0.8077	0.7950	1.28%
2	5	5	15	600	100	5×10^{-3}	10^{-4}	0.7979	0.7887	0.7964	0.7872	0.91%
3	5	5	15	1000	100	10^{-2}	0	0.8149	0.8051	0.8137	0.8041	0.98%

Tabella 8: Risultati con ottimizzatore SGD e attivazione *tanh* al variare del learning rate.

6.6 Esperimento 1 (Tanh)

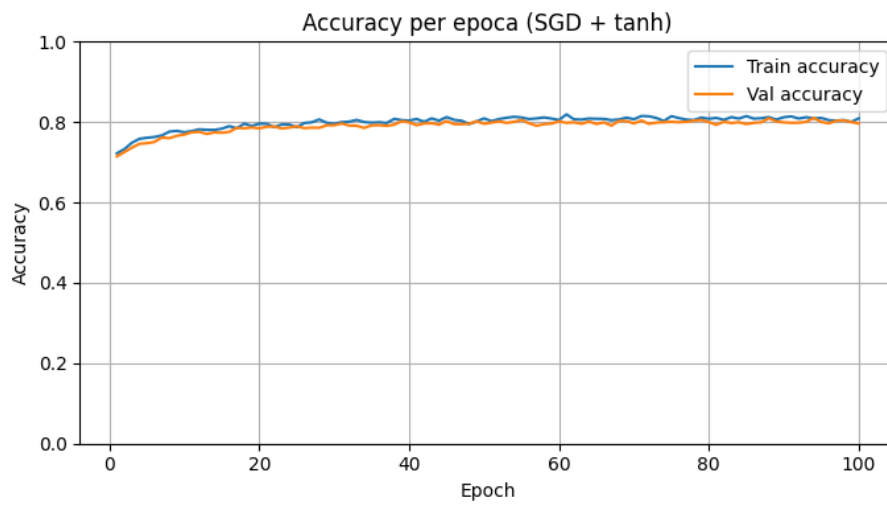


Figura 100: Esperimento 1 (Tanh): andamento dell'accuracy (train vs validation).

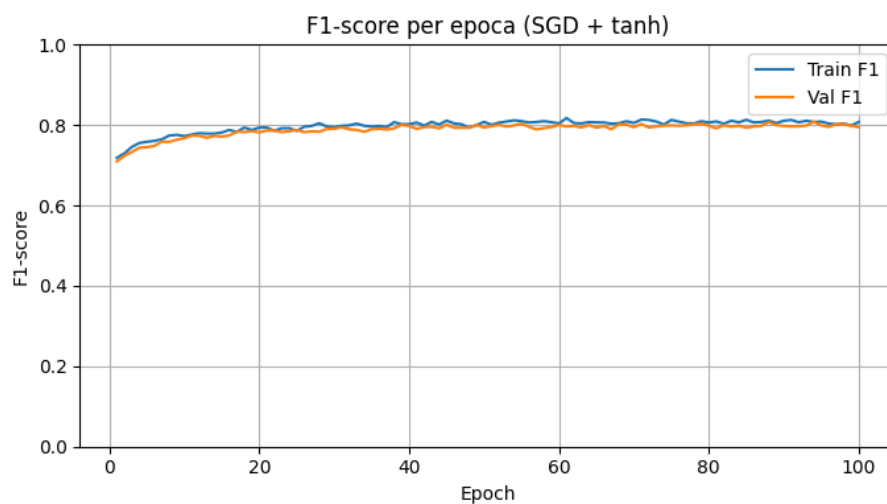


Figura 101: Esperimento 1 (Tanh): andamento del macro F1-score (train vs validation).

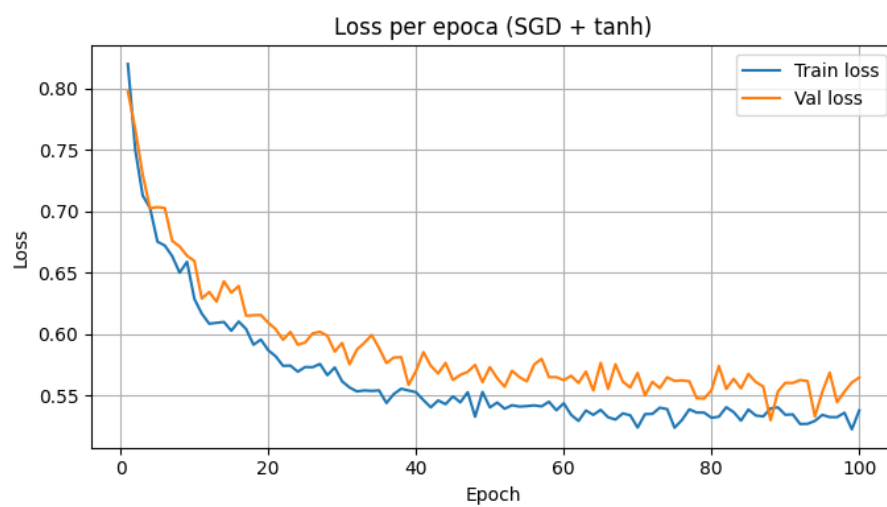


Figura 102: Esperimento 1 (Tanh): andamento della loss (train vs validation).

[Torna alla tabella dei risultati](#)

6.7 Esperimento 2 (Tanh)

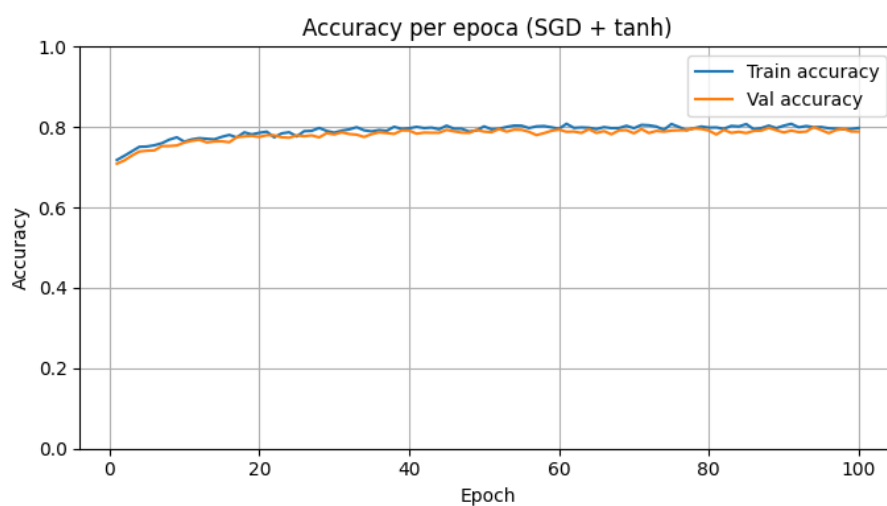


Figura 103: Esperimento 2 (Tanh): andamento dell'accuracy (train vs validation).

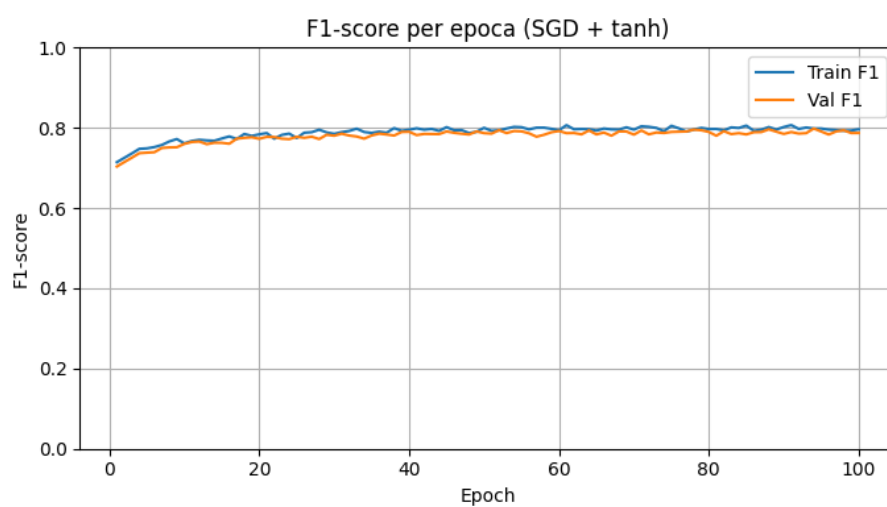


Figura 104: Esperimento 2 (Tanh): andamento del macro F1-score (train vs validation).

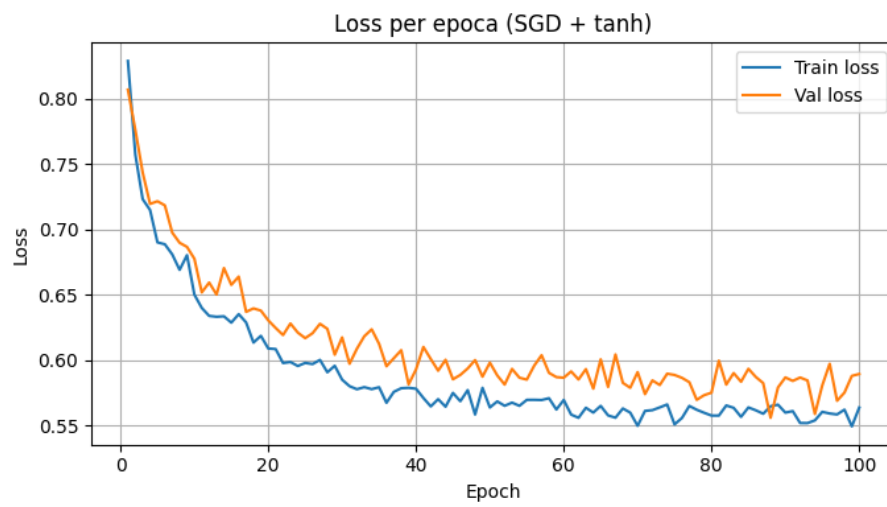


Figura 105: Esperimento 2 (Tanh): andamento della loss (train vs validation).

[Torna alla tabella dei risultati](#)

6.8 Esperimento 3 (Tanh)

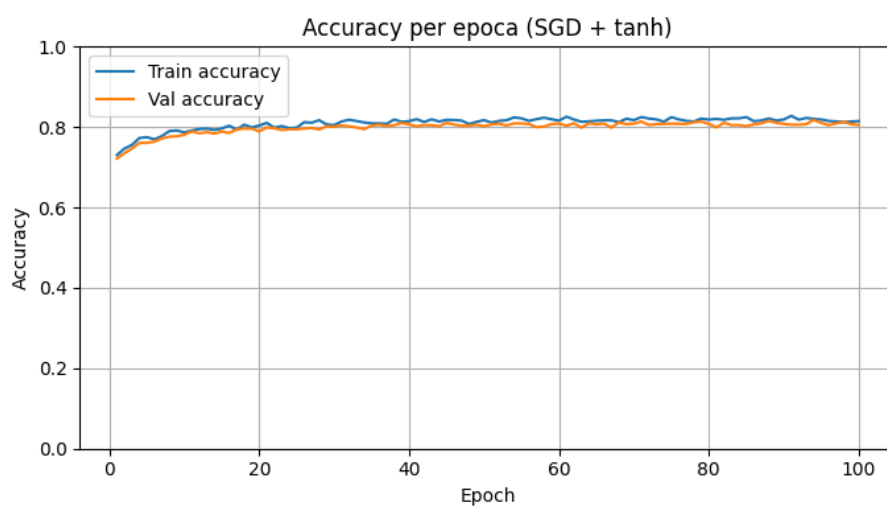


Figura 106: Esperimento 3 (Tanh): andamento dell'accuracy (train vs validation).

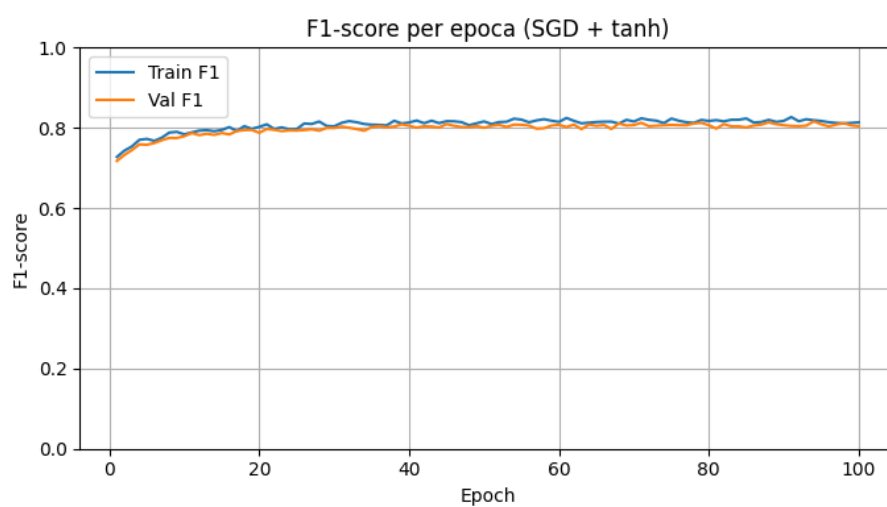


Figura 107: Esperimento 3 (Tanh): andamento del macro F1-score (train vs validation).

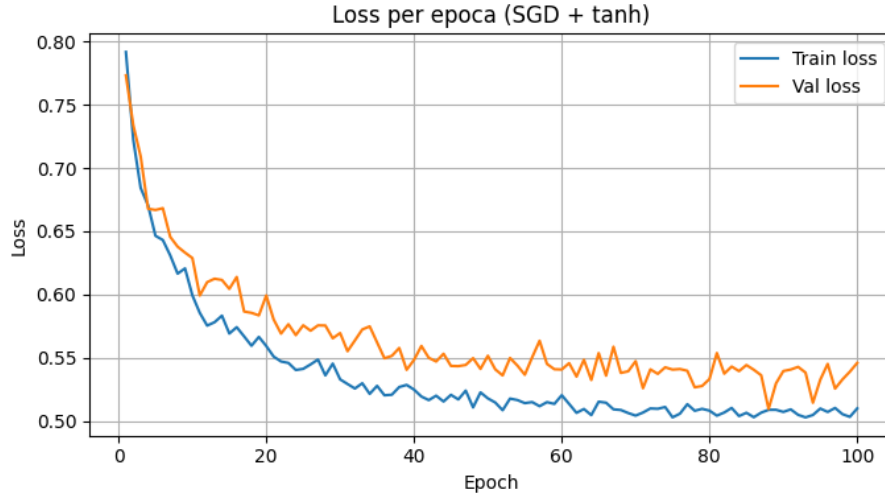


Figura 108: Esperimento 3 (Tanh): andamento della loss (train vs validation).

[Torna alla tabella dei risultati](#)

6.9 Confronto risultati

Exp	lr	weight decay	Episodi	Epoche	Train Acc (tanh)	Train Acc (ReLU)	Val Acc (tanh)	Val Acc (ReLU)	Gap tanh	Gap ReLU
1	10^{-2}	10^{-4}	600	100	0.8092	0.8230	0.7964	0.8100	1.28%	1.28%
2	5×10^{-3}	10^{-4}	600	100	0.7979	0.8180	0.7887	0.8070	0.91%	1.16%
3	10^{-2}	0	1000	100	0.8149	0.8300	0.8051	0.8120	0.98%	1.85%

Tabella 9: Confronto diretto tra attivazione *tanh* e ReLU nella head MLP del Prototypical Network (cosine similarity) a parità di learning rate e weight decay.

Dai risultati emerge un confronto diretto tra le attivazioni tanh e ReLU nella head MLP del Prototypical Network, mantenendo fissi ottimizzatore (SGD), learning rate, weight decay e l'uso del learning rate scheduler. In tutti gli esperimenti, la ReLU mostra valori di training accuracy leggermente superiori rispetto alla tanh, indicando una maggiore capacità di adattamento ai dati di addestramento. Questo comportamento è coerente con la natura non saturante della ReLU, che facilita l'ottimizzazione e consente una maggiore flessibilità nella modellazione delle rappresentazioni. Per quanto riguarda le prestazioni di validazione, le differenze tra le due attivazioni risultano più contenute: nei primi due casi la ReLU mantiene un lieve vantaggio, mentre nel terzo le prestazioni sono comparabili. Tuttavia, analizzando il gap train-validation, si osserva che la tanh tende a produrre un divario più ridotto rispetto alla ReLU nella maggior parte degli esperimenti.

Questo suggerisce che, sebbene la ReLU favorisca una migliore ottimizzazione sul training set, la tanh introduce un effetto di

regolarizzazione implicita dovuto alla saturazione delle attivazioni, limitando l'adattamento eccessivo ai dati di addestramento. Di conseguenza, la \tanh mostra una maggiore stabilità in termini di generalizzazione, mentre la ReLU presenta un gap più ampio, indice di una maggiore propensione all'overfitting nel contesto episodico few-shot.

6.9.1 Esperimento 1

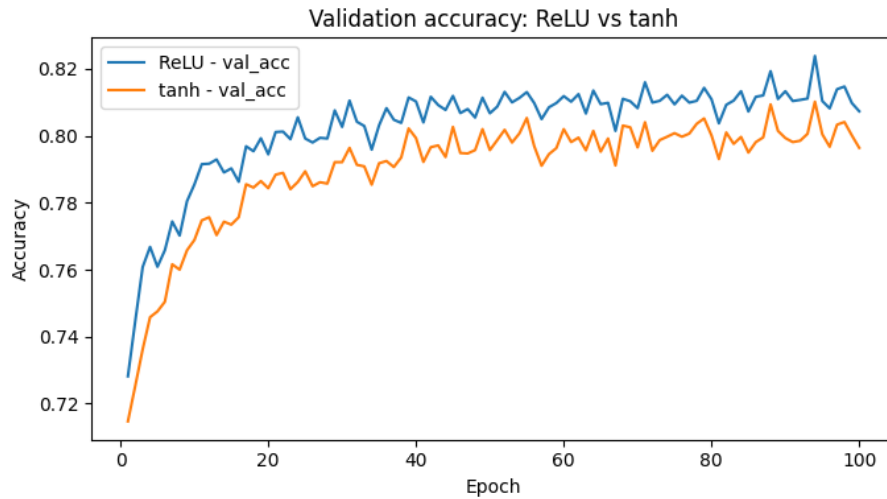


Figura 109: Esperimento 1: confronto dell'accuracy di validazione tra ReLU e Tanh.

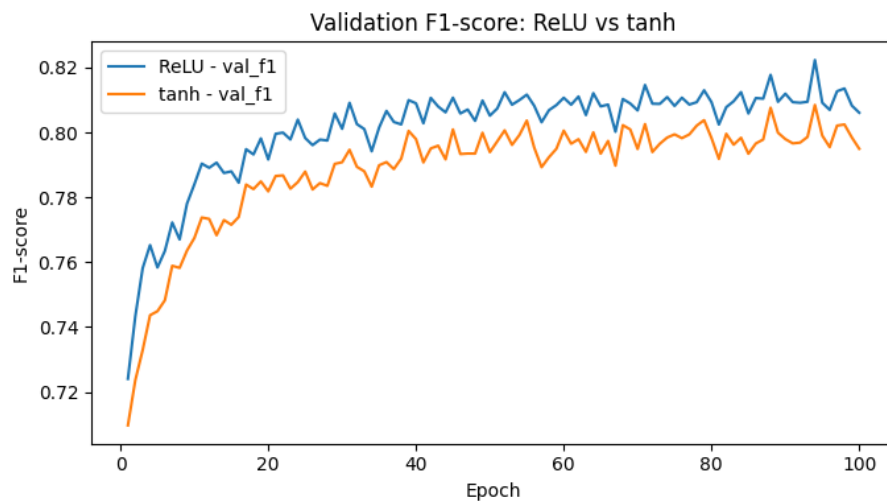


Figura 110: Esperimento 1: confronto del macro F1-score di validazione tra ReLU e Tanh.

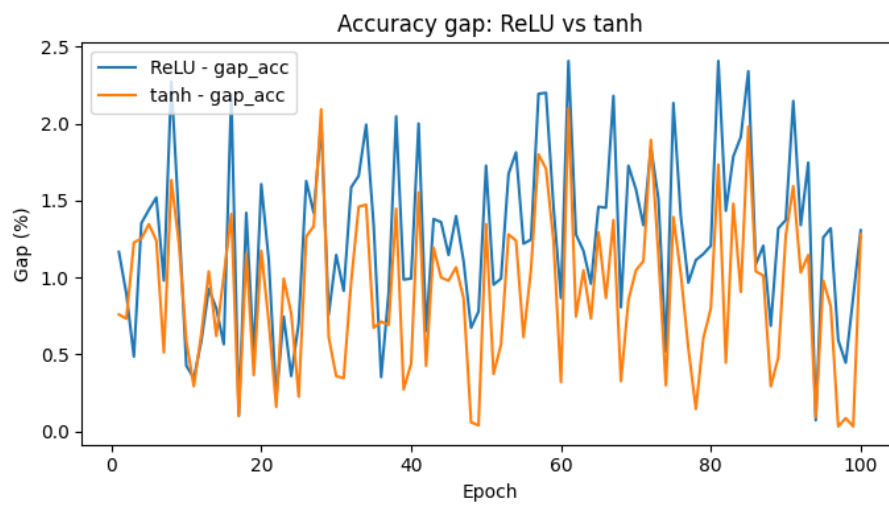


Figura 111: Esperimento 1: confronto del gap train-validation tra ReLU e Tanh.

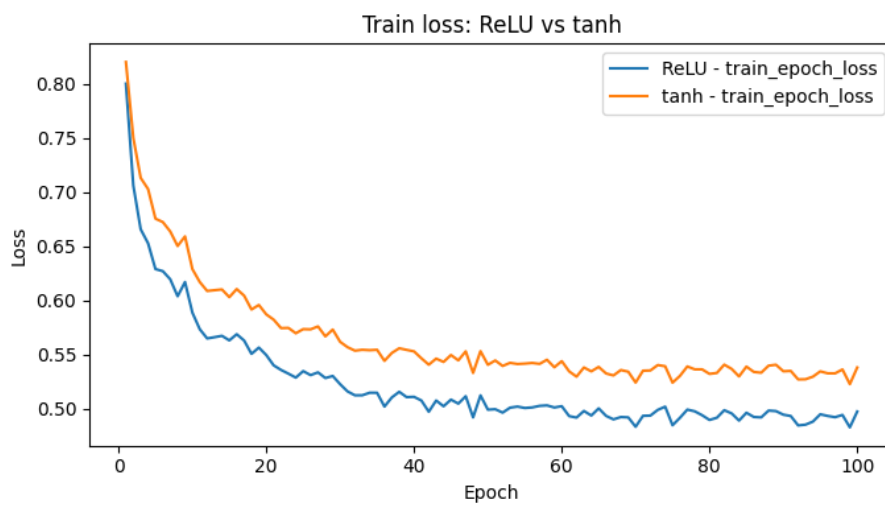


Figura 112: Esperimento 1: confronto della loss di training tra ReLU e Tanh.

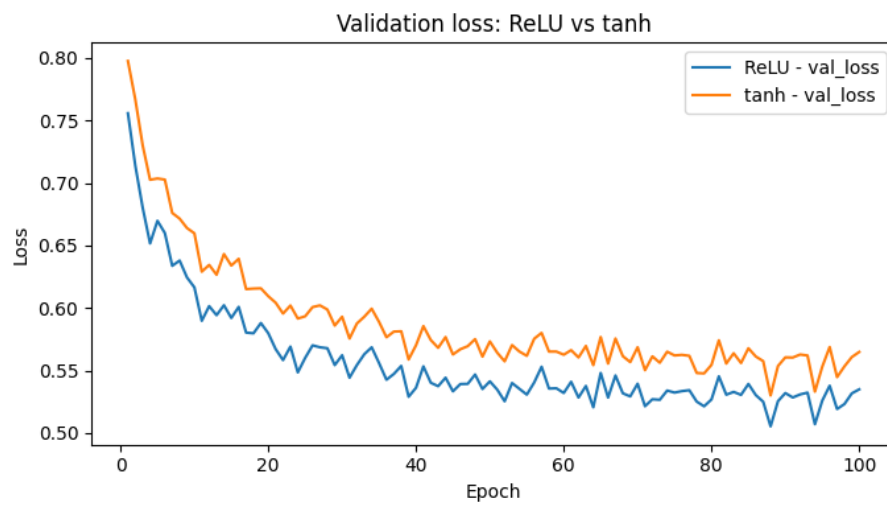


Figura 113: Esperimento 1: confronto della loss di validazione tra ReLU e Tanh.

[*Torna alla tabella dei risultati*](#)

6.9.2 Esperimento 2

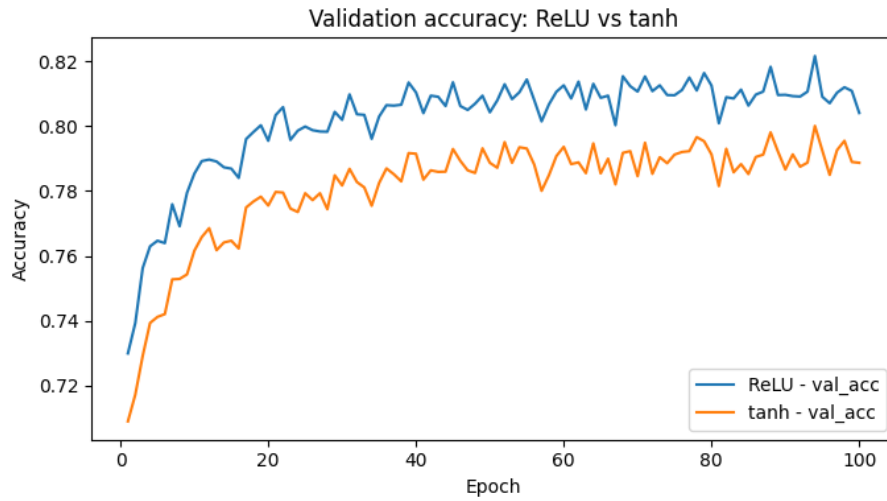


Figura 114: Esperimento 2: confronto dell'accuracy di validazione tra ReLU e Tanh.

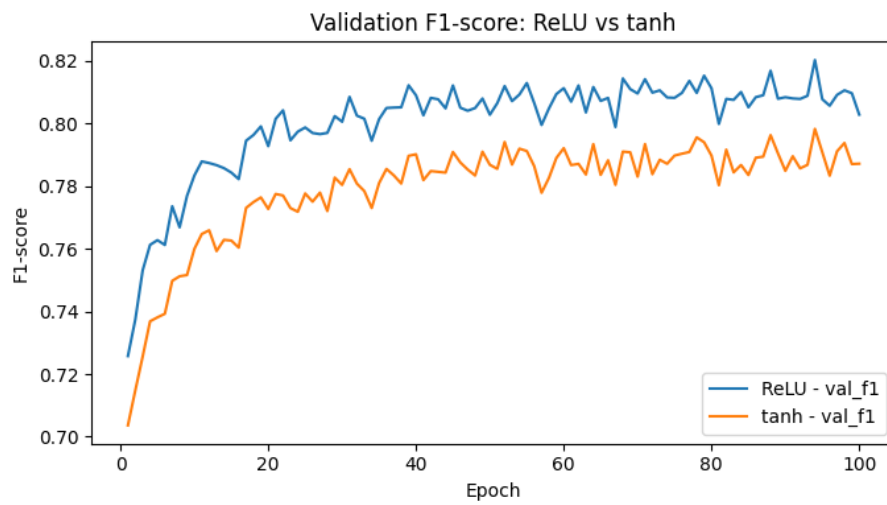


Figura 115: Esperimento 2: confronto del macro F1-score di validazione tra ReLU e Tanh.

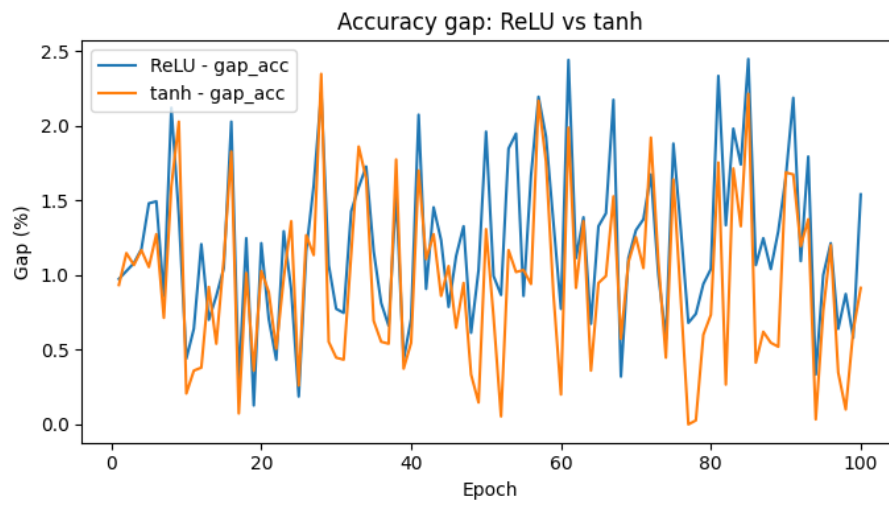


Figura 116: Esperimento 2: confronto del gap train-validation tra ReLU e Tanh.

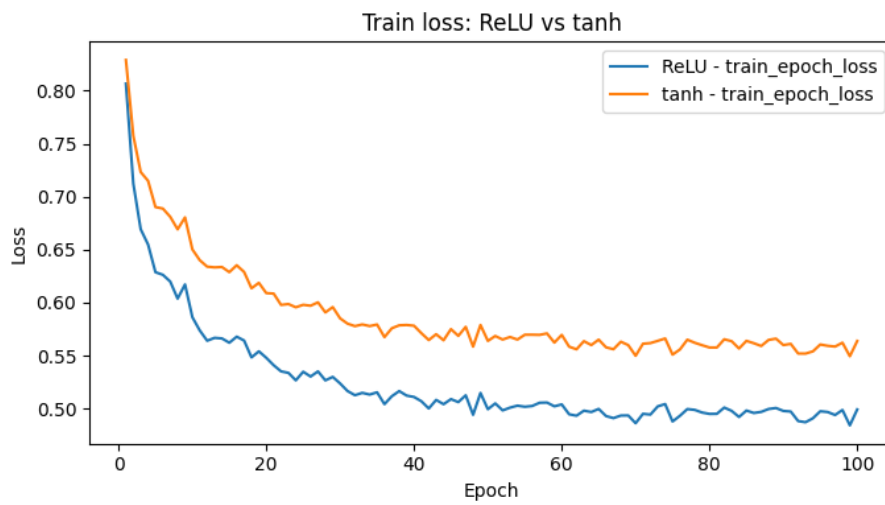


Figura 117: Esperimento 2: confronto della loss di training tra ReLU e Tanh.

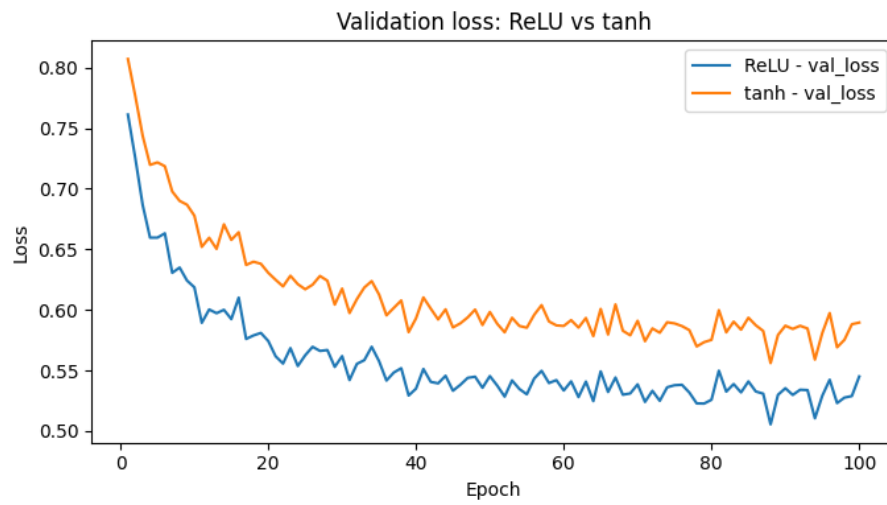


Figura 118: Esperimento 2: confronto della loss di validazione tra ReLU e Tanh.

[Torna alla tabella dei risultati](#)

6.9.3 Esperimento 3

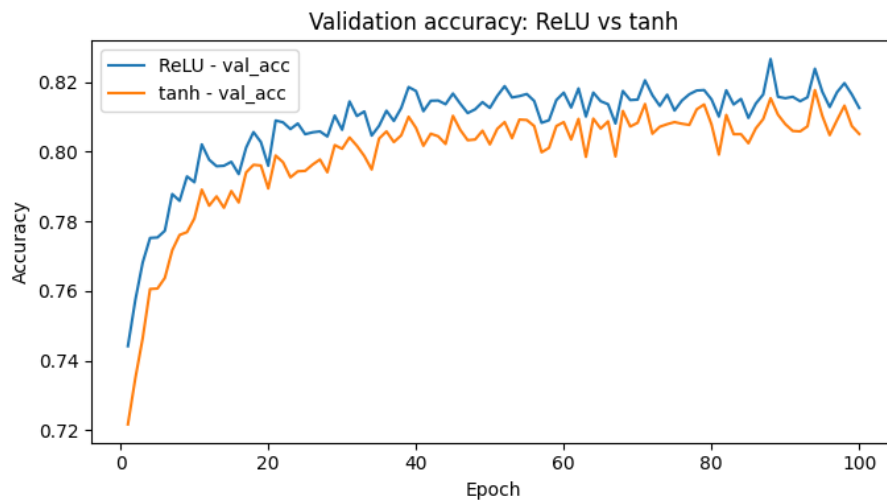


Figura 119: Esperimento 3: confronto dell'accuracy di validazione tra ReLU e Tanh.

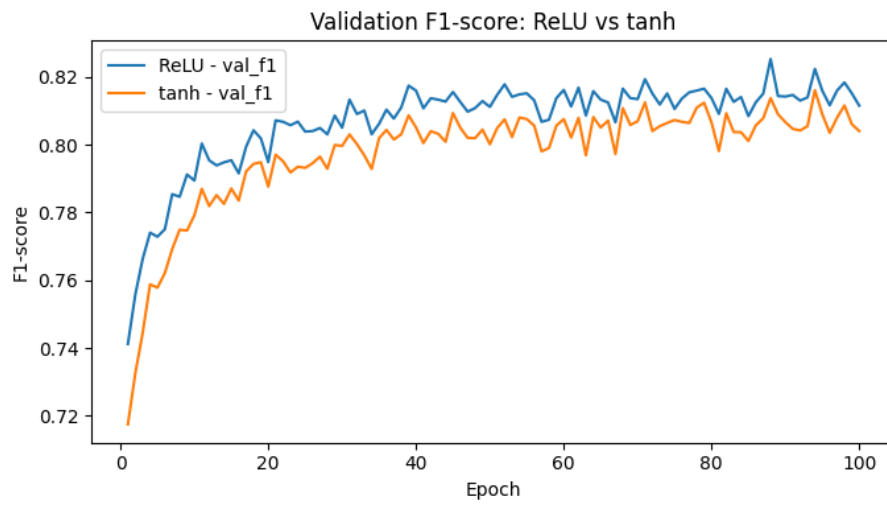


Figura 120: Esperimento 3: confronto del macro F1-score di validazione tra ReLU e Tanh.

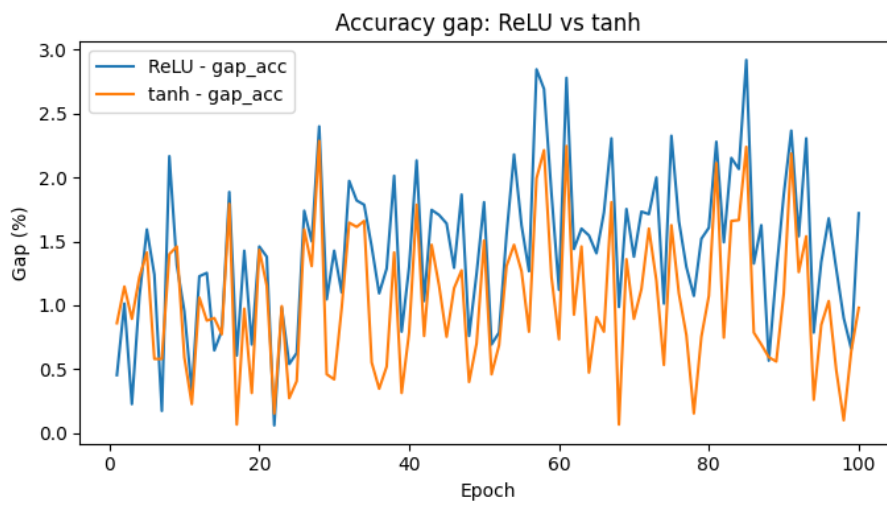


Figura 121: Esperimento 3: confronto del gap train-validation tra ReLU e Tanh.

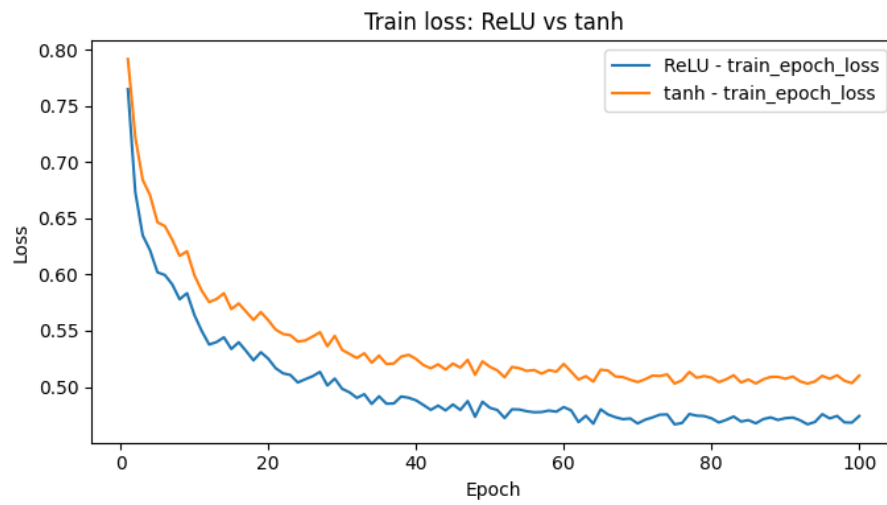


Figura 122: Esperimento 3: confronto della loss di training tra ReLU e Tanh.

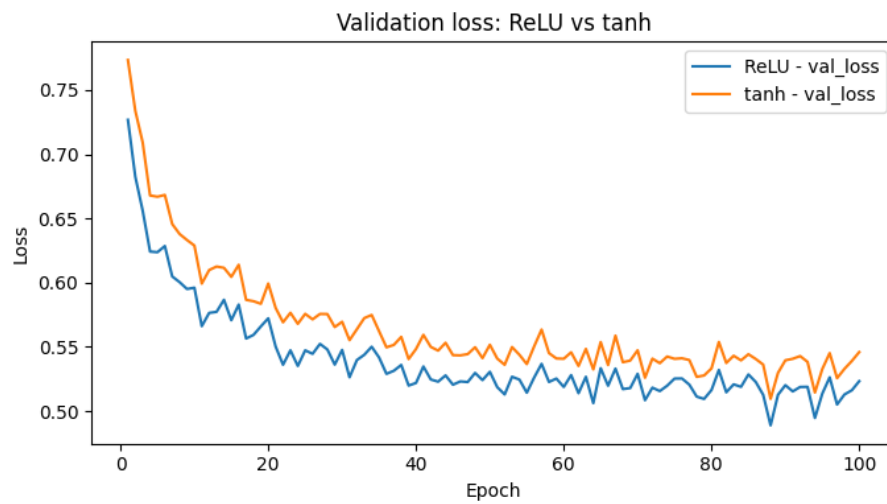


Figura 123: Esperimento 3: confronto della loss di validazione tra ReLU e Tanh.

[Torna alla tabella dei risultati](#)

7 Selezione del modello finale

In questa sezione viene effettuata la selezione del modello finale tra le diverse configurazioni sperimentali analizzate. Per rendere il confronto equo e coerente il confronto è limitato agli esperimenti condotti con la stessa configurazione episodica (5-way, 5-shot, 15 query), mantenendo quindi costante la difficoltà del task. La scelta del modello finale è presa sui valori di F1-score sul validation e sul gap tra training e validation per garantire un buon compromesso tra accuratezza e capacità di generalizzazione.

ID	Configurazione del modello	Val F1	Gap (%)
1	ProtoNet, head lineare, distanza cosine	≈ 0.754	≈ 0.16
2	ProtoNet, head lineare, distanza euclidea	≈ 0.721	≈ 0.15
3	ProtoNet, head non lineare (ReLU), distanza cosine	≈ 0.812	≈ 1.23
4	ProtoNet, head non lineare (ReLU), distanza euclidea	≈ 0.794	≈ 0.41
5	ProtoNet, head non lineare (tanh), distanza cosine	≈ 0.813	≈ 0.89
6	ProtoNet + SGD + StepLR, head ReLU, distanza cosine	≈ 0.809	≈ 1.28
7	ProtoNet + SGD + StepLR, head tanh, distanza cosine	≈ 0.804	≈ 0.98

Tabella 10: Tabella riepilogativa delle configurazioni considerate per la selezione del modello finale. Tutti gli esperimenti sono condotti in configurazione 5-way, 5-shot, 15 query.

La scelta del modello finale è presa sui valori di F1-score sul validation e sul gap tra training e validation per garantire un buon compromesso tra accuratezza e capacità di generalizzazione. Osservando i risultati ottenuti dalle diverse configurazioni è possibile selezionare come modello finale il modello 5, ovvero il modello con head non lineare (tanh) e distanza cosine, che presenta il valore di F1-score sul validation set più elevato, a fronte di un gap train-validation contenuto. Una volta individuata la configurazione finale del modello, si procede alla sua valutazione sul test set, che non è stato utilizzato in alcuna fase di addestramento o selezione. In particolare, vengono calcolate le metriche di performance sul test set e analizzata la matrice di confusione, al fine di valutare il comportamento del modello sulle singole classi e individuare eventuali pattern di errore.

7.1 Valutazione del modello finale

Una volta selezionata la configurazione finale del modello sulla base delle prestazioni sul validation set passiamo al test sul test set per fornire una stima affidabile della capacità di generalizzazione.

Oltre alle metriche di performance globali verrà successivamente analizzata la matrice di confusione sul test set con l'obiettivo di studiare il comportamento del modello sulle singole classi e individuare eventuali pattern sistematici di errore.

7.1.1 Valori sul Test

Loss	Accuracy	F1-score
0.5425	0.8107	0.8095

Tabella 11: Risultati di performance del modello finale sul test set.

I risultati sul test set mostrano che il modello finale mantiene buone capacità di generalizzazione, scendendo nel particolare l'accuracy pari a **0.8107** e l'F1 score pari a **0.8095** ci indicano un equilibrio soddisfacente tra Precision e Recall suggerendo che il modello non è sbilanciato verso una singola classe. La loss sul test set (0.5425) è coerente con le metriche citate precedentemente e non evidenzia fenomeni evidenti di degrado delle prestazioni rispetto alla fase di validazione.

Nel complesso questi risultati confermano la stabilità del modello e la sua efficacia nel contesto di classificazione considerato.

7.1.2 Confusion Matrix

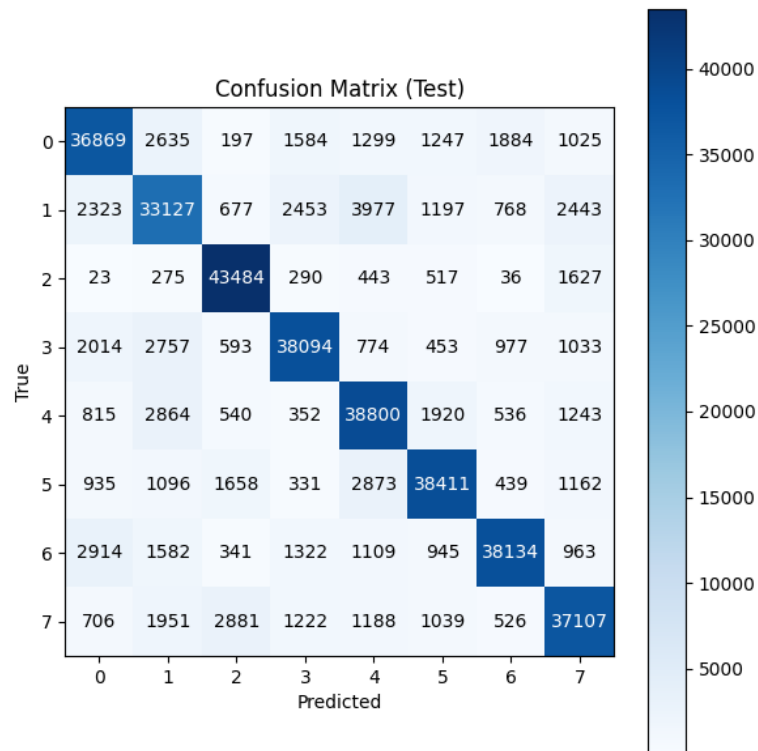


Figura 124: Matrice di confusione del modello finale sul test set.

La confusion matrix sul test set mostra una buona concentrazione dei campioni sulla diagonale indicando che la maggior parte delle osservazioni viene identificata correttamente. In particolare le classi dalla 2 alla 7 presentano un numero elevato di veri positivi. Inoltre notiamo confusioni sistematiche tra alcune coppie di classi come tra la classe e la classe 4 o la classe 0 con le classi 1,3 e 6. Notiamo anche che per alcune classi la distribuzione degli errori è particolarmente diffusa indicando quindi una minore robustezza del confine decisionale rispetto ad altre classi più facilmente separabili.