

Progetto Deep Learning 23/24

Francesco Bottalico
f.bottalico15@studenti.uniba.it

Abstract

Questo progetto esplora l'efficacia del metodo di *Textual Inversion* nel replicare lo stile di artisti famosi utilizzando *Stable Diffusion*. Attraverso il fine-tuning su un dataset di opere d'arte di un modello pre-addestrato, sono state valutate diverse varianti del metodo, inclusi moduli basati su meccanismi di attenzione ispirati a recenti lavori e viene presentata una nuova architettura basata su *Mixture of Experts*. I risultati mostrano che la *Textual Inversion* migliora significativamente la capacità del modello di generare immagini coerenti con lo stile di un artista rispetto al semplice *Prompt Engineering*, con il modulo *Mixture of Experts* che ottiene i migliori risultati in termini di accuratezza.

1 Introduzione

Tra le varie tecniche di generazione di immagini, la **Stable Diffusion** [5] si distingue per la sua capacità di generare immagini altamente realistiche e fedeli al prompt testuale. Questo progetto si concentra sull'utilizzo del metodo di **Textual Inversion** [2] per catturare e riprodurre fedelmente gli stili distintivi di vari artisti. La Stable Diffusion permette di sintetizzare nuove immagini a partire da una descrizione testuale. Questo approccio ha trovato applicazioni in numerosi settori, dalla creazione di contenuti visivi personalizzati alla simulazione artistica e al restauro digitale. Il metodo di Textual Inversion permette di adattare modelli di generazione di immagini per replicare con precisione lo stile di un determinato artista attraverso un processo di fine-tuning su un corpus di opere esistenti. La Textual Inversion funziona invertendo il processo di generazione delle immagini: invece di partire da un testo per generare un'immagine, questo metodo permette di partire da un'immagine per ottenere una rappresentazione testuale (pseudo-token) che cattura le caratteristiche stilistiche distintive dell'opera d'arte. Questa rappresentazione testuale può poi essere utilizzata nel prompt per generare nuove immagini che mantengono lo stesso stile artistico. Questo progetto mira a esplorare e valutare l'efficacia del metodo di Textual Inversion nel replicare gli stili di artisti famosi, utilizzando anche metriche quantitative basate sul modello **CLIP** [3]. Verranno presentati i risultati di vari esperimenti condotti utilizzando un dataset di artisti con un piccolo sottoinsieme delle loro opere d'arte, utilizzando diverse tecniche di Textual Inversion ispirate a vari lavori recenti.

2 Related Works

Nell'ambito dello style transfer, i modelli basati su Stable Diffusion possono replicare lo stile di artisti famosi semplicemente attraverso prompt engineering, ovvero includendo nel

testo in input al modello lo stile che si vorrebbe replicare. Se però volessimo dei risultati più precisi e che funzionino bene anche per artisti di minor fama, potremmo utilizzare il metodo di Textual Inversion [2] che permette di apprendere uno *pseudo-token* nello spazio vettoriale dei token che codifichi lo stile di un artista. Diversi miglioramenti di questo metodo sono stati proposti successivamente, i quali utilizzano un modulo complesso (una rete neurale) per apprendere lo *pseudo-token*. [7] propone un modulo basato su un meccanismo di attenzione per apprendere una rappresentazione di stile più accurata. In questo progetto, oltre al metodo standard, verranno valutati due moduli di Textual Inversion, uno ispirato all'opera appena citata ed uno ispirato ad un'architettura ultimamente molto diffusa nei Large Language Models ed utilizzabile anche nei Vision Transformers [4]: la **Mixture of Experts** (MoE). Una Mixture of Experts è una rete neurale dove sono presenti più reti indipendenti tra loro, gli experts, ed una routing function viene appresa per indirizzare un input verso uno o più esperti. L'idea è quella di avere un ciascun esperto specializzato su una parte specifica dei dati come in una sorta di ensemble.

3 Task

Il task è leggermente diverso rispetto al classico style transfer, infatti in quest'ultimo abbiamo un'immagine di stile, una di contenuto ed il nostro scopo è fondere stile e contenuto in un'unica immagine. Nel nostro caso abbiamo un insieme di artisti ed alcune loro opere e vogliamo codificare lo stile di un artista in uno specifico embedding, il quale poi potrà essere riutilizzato durante la fase di generazione di una immagine per condizionarne lo stile con quello dell'artista. Il contenuto dell'immagine generata non verrà preso da un'immagine in input, bensì verrà generata a partire dal prompt testuale. Data la differenza significativa con lo style transfer, il modulo di Textual Inversion presentato in [7] verrà adattato al nostro task e migliorato dopo aver analizzato alcune sue possibili limitazioni.

4 Dataset

Il dataset utilizzato è stato ottenuto a partire da **ArtGraph** [1], dal quale sono stati scelti dieci artisti di diversi livelli di fama. La scelta di avere artisti di diversa popolarità è stata fatta per valutare la capacità del modello di Stable Diffusion nel replicare lo stile di un artista solamente modificando il prompt durante la generazione dell'immagine: ci aspettiamo che questo approccio funzioni bene per gli artisti famosi, meno per gli altri. Usando la Textual Inversion invece dovremmo ottenere dei buoni risultati per tutti. Per ciascun artista sono state scelte quattro opere appartenenti allo stesso stile (attributo **artgraph:hasStyle** che collega opera e stile), badando di scegliere opere di stile diverso per artisti differenti. Avremo quindi dieci stili diversi, uno per artista.

Gli artisti scelti sono i seguenti: Claude Monet, Edvard Munch, Erte, Eyvind Earle, Gustav Klimt, Ilya Mashkov, Luca Giordano, Mabuse, Patrick Pietropoli, Paul Cezanne.

5 Esperimenti Preliminari

Prima di sperimentare la Textual Inversion per codificare lo stile di un artista, sono stati valutati altri approcci che non richiedono l'utilizzo di fine-tuning. In particolare, l'idea

iniziale era quella di impiegare il Variational Autoencoder di un modello di Stable Diffusion pre-addestrato per codificare le opere di un artista, calcolare il centroide ed utilizzare quest'ultimo come codifica dello stile. Questo poi verrebbe usato come punto di partenza durante la fase di generazione di un'immagine, con l'aggiunta di una certa quantità di rumore, invece di utilizzare solamente rumore casualmente generato. In figura 1 vediamo un esempio di centroide decodificato: le due immagini a sinistra sono state codificate nello spazio latente, è stato calcolato il centroide che poi è stato decodificato (immagine a destra). Possiamo notare come lo stile ed alcuni elementi delle immagini iniziali vengano mantenuti. Inoltre è bene precisare che questo approccio funziona meglio se si prendono poche immagini per il calcolo del centroide (2-3).

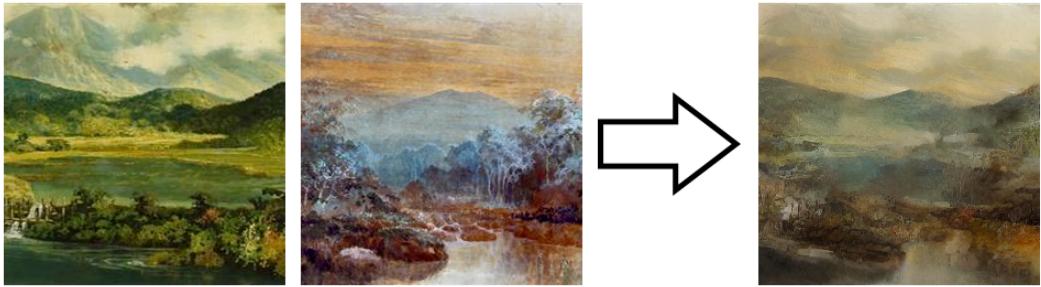


Figure 1: Sulla destra il centroide decodificato ottenuto a partire dalle due opere a sinistra.

Problemi Il problema principale risiede nella fase di generazione delle immagini: utilizzare il centroide come punto di partenza per la diffusion non riesce a trasferire lo stile dell'artista in maniera effettiva, come si vede in figura 2. Per quanto i colori vengano più o meno mantenuti, lo stile della pennellata e le texture originali si perdono completamente.



Figure 2: Immagine generata a partire dal centroide e immagini originali.

6 Metodologia

Gli esperimenti successivi sono stati eseguiti impiegando quattro metodi differenti:

- **Prompt Engineering (PE):** lo stile da replicare viene inserito nel prompt (es. “... in the style of Claude Monet”).
- **Textual Inversion (TI):** metodologia standard di Textual Inversion senza l’uso di moduli complessi.
- **Textual Inversion - Attention (TI-Att):** modulo di Textual Inversion basato su meccanismi di attenzione, variante di [7] adattata al nostro specifico task.
- **Textual Inversion - MoE (TI-MoE):** modulo di Textual Inversion basato su Mixture of Experts e presentato nella sezione 6.3.

I diversi approcci, eccezion fatta per il primo che non richiede fine-tuning, vengono implementati utilizzando un modello di Stable Diffusion pre-allenato. Il processo di diffusion avviene in uno spazio latente appreso durante il training da un VAE, utilizza una U-Net come modello di denoising ed impiega il text encoder di CLIP per codificare il prompt testuale.

6.1 Textual Inversion

Il primo approccio consiste nell’apprendere lo stile di un artista nello spazio vettoriale dei token, ovvero il primo layer (embedding layer) del text encoder. Lo stile quindi, alla fine del fine-tuning, sarà associato ad un nuovo pseudo-token. Tutti i pesi, del VAE, della U-Net e del text encoder, vengono tenuti bloccati a parte quelli associati allo pseudo-token (Figura 3).

Il processo di training è speculare a quello utilizzato per allenare un modello di diffusion,

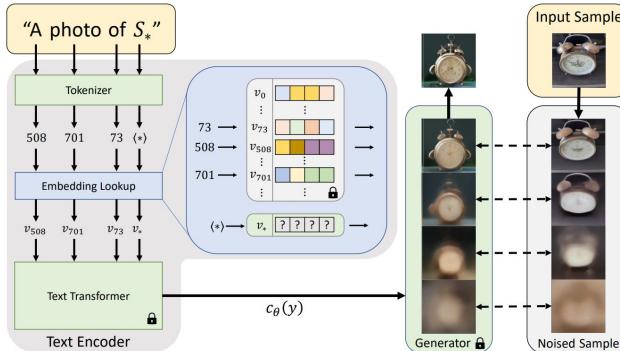


Figure 3: Textual Inversion training.

la loss è la stessa utilizzata nel paper di Stable Diffusion (meno la componente associata all’ottimizzazione del VAE).

$$\mathcal{L} := \|\epsilon - \epsilon_\theta(z_t, t, c_\theta(y))\|_2^2 \quad (1)$$

Dove $\epsilon \sim \mathcal{N}(0, 1)$, ϵ_θ è la predizione di U-Net, t il timestep utilizzato nella fase di forward diffusion, z_t l’immagine in input con l’aggiunta del rumore fino al timestep t , c_θ è il text encoder e y il prompt testuale.

6.2 Textual Inversion - Attention

Il processo di fine-tuning per questo approccio è identico a quello appena mostrato, con la differenza che l'encoding dell'artista non viene appreso nell'embedding layer del text encoder, dove vengono inizializzati dei pesi e modificati durante la fase di ottimizzazione, bensì è ottenuto come output di un'altra rete neurale (che chiamiamo **Inversion Module**). Questo viene poi utilizzato come embedding dello pseudo-token associato all'artista, in input al text encoder (Figura 4). L'input dell'Inversion Module è l'embedding dell'immagine stile ottenuta utilizzando l'image encoder di CLIP.

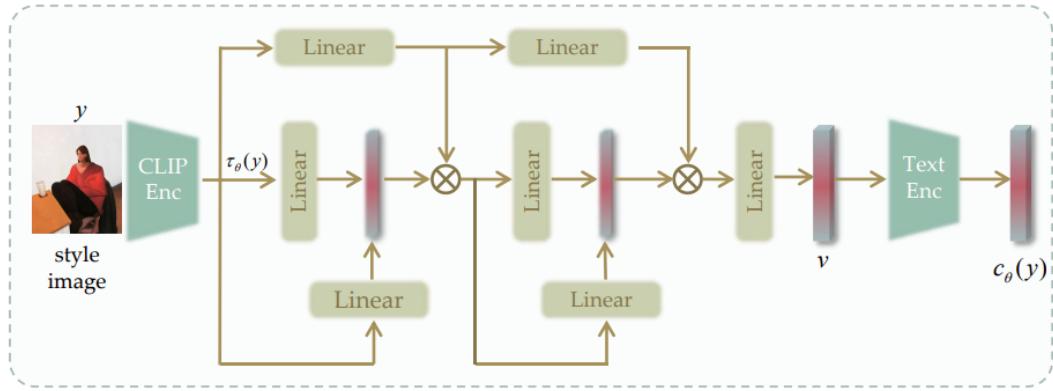


Figure 4: Inversion Module basato su un doppio layer di attenzione.

La rete neurale presentata in [7] è costituita da due attention layer seguiti da un layer lineare.

$$Q_i = W_Q^{(i)} \cdot v_i, K = W_K^{(i)} \cdot \tau_\theta(y), V = W_V^{(i)} \cdot \tau_\theta(y)$$

$$v_{i+1} = \text{Attention}(Q_i, K, V)$$

Dove $v_0 = \tau_\theta(y)$, y è l'immagine di input e *Attention* è la funzione di multi-head attention come definita in [6]. $\tau_\theta(y)$ è l'image encoder di CLIP.

Possibili limitazioni L'implementazione di questo modulo come definita in origine ha principalmente due problemi:

1. Nel paper originale l'immagine di stile è una, questo comporta che la funzione di attention viene applicata su una sequenza di lunghezza pari ad 1, che ne limita fortemente l'utilità facendola diventare di fatto una funzione lineare. Infatti se la lunghezza della sequenza è 1 si può facilmente verificare che:

$$\text{Attention}(QW_Q, KW_K, VW_V)W_O \iff VW_VW_O$$

2. Tra il secondo attention layer ed il layer lineare non vi è alcuna funzione di attivazione non lineare, questo potrebbe portare ad una perdita di performance.

Risoluzioni I due problemi sopracitati sono stati risolti nei seguenti modi:

1. Utilizziamo una sequenza di lunghezza 4 come input dell'attention layer, ogni elemento della sequenza è un'opera dell'artista codificata utilizzando l'encoder di CLIP. Aggiungiamo inoltre come primo elemento della sequenza un vettore di pesi allenabili, il quale poi sarà l'unico utilizzato come embedding finale dello stile. Per fare questo è stata presa ispirazione da modelli come BERT o i Vision Transformer che utilizzano un token fittizio [CLS] per la classificazione del testo/immagine.
2. Il primo elemento della sequenza in output al doppio attention layer viene dato in input al layer lineare non prima di aver applicato una non linearità, in questo caso la **GELU**. Questo, come spiegato, sarà utilizzato come embedding dello stile.

6.3 Textual Inversion - Mixture of Experts

In questo caso l'Inversion Module implementa un'architettura basata su **Mixture of Experts** ispirata a [4]. In questo paper viene presentata l'architettura come sostituto del blocco FFN (feedforward network) all'interno dei Transformer: invece di avere un singolo FFN dopo l'attention layer al quale vengono dati in input tutti gli embedding della sequenza, avremo n FFN indipendenti tra loro ed una *routing function* che ne seleziona k (con $k < n$) per ogni elemento della sequenza. Dopodichè ciascuno di essi viene dato in input ai k FFNs assegnategli e gli output sono combinati attraverso una combinazione lineare con i coefficienti generati dalla *routing function*.

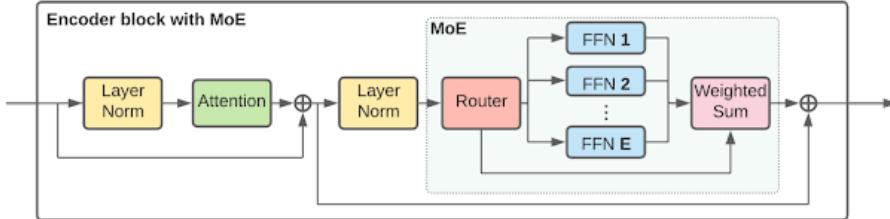


Figure 5: Blocco MoE in un Vision Transformer.

Il blocco MoE in figura 5 viene utilizzato come base dell'Inversion Module, in input avremo sempre la sequenza di 4 immagini dell'artista codificate dall'image encoder (un Vision Transformer) di CLIP. I quattro embedding in output al MoE vengono poi compressi in uno utilizzando la concatenazione seguita da un layer lineare al fine di ottenere una rappresentazione unica per lo stile dell'artista. L'idea è che ciascun esperto possa specializzarsi in un tipo di immagini differente, per esempio se ci sono opere rappresentanti soggetti diversi, ciascun esperto potrebbe perfezionarsi per un soggetto specifico.

Implementazione I FFNs $f_i(x)$ sono implementati con due layer lineari intermezzati da una funzione non lineare: $f_i(x) = W_2\sigma_{gelu}(W_1x)$.

La routing function invece è implementata come $g(x) = TOP_k(\text{softmax}(Wx + \epsilon))$, dove $\epsilon \sim \mathcal{N}(0; \frac{1}{E^2})$, E è il numero di esperti e W è una matrice di pesi di dimensione $E \times D$ con D dimensione degli embedding nella sequenza.

Loss La loss utilizzata durante il training è sempre quella mostrata nell’equazione 1. In aggiunta a questa, vengono proposte altre due loss:

1. **Importance Loss:** per incentivare l’utilizzo di tutti gli esperti in maniera bilanciata, ovvero far sì che i coefficienti in output alla routing function siano bilanciati tra i vari esperti.
2. **Load Loss:** per garantire l’utilizzo di diversi esperti per elementi diversi della sequenza. Senza questa loss rischieremmo che la routing function selezionasse sempre gli stessi esperti a danno degli altri.

La loss ausiliaria quindi è definita come:

$$\mathcal{L}_{aux}(X) = \frac{1}{2}\mathcal{L}_{imp}(X) + \frac{1}{2}\mathcal{L}_{load}(X) \quad (2)$$

mentre la loss complessiva sarà $\mathcal{L} + \lambda\mathcal{L}_{aux}$, dove λ è un coefficiente di importanza per la loss ausiliaria.

7 Esperimenti

Verranno valutate le quattro configurazioni definite nella sezione 6, per ciascuna di esse effettuiamo 1000 step di ottimizzazione, con **batch size** di 4 ed un **learning rate** pari a 0.005 per la Textual Inversion standard, mentre per le altre due varianti utilizziamo un learning rate pari a 0.001. Per il modulo implementato con MoE utilizziamo un totale di 4 esperti e $k = 2$, mentre il coefficiente per la loss ausiliaria $\lambda = 0.01$. È bene ricordare che i pesi del VAE, della U-Net, del text encoder e dell’image encoder sono bloccati, a differenza di quelli dell’Inversion Module. I prompt utilizzati durante la fase di training associati a ciascuna immagine sono scelti casualmente ad ogni iterazione da una lista di 19 frasi e sono tutti varianti di frasi del tipo: “*a painting in the style of <artist-name>*”, dove **<artist-name>** è lo pseudo-token associato allo stile dell’artista.

7.1 Preprocessing

Per ciascun artista le quattro immagini di training vengono ripetute per 100 volte in un’epoca, con una probabilità di 0.5 di venire specchiate. In 10 epoche avremo $10 \cdot 400 = 4000$ immagini di training, uniti ad una batch size pari a 4 otteniamo i nostri $4000/4 = 1000$ step di ottimizzazione. Inoltre le immagini vengono ingrandite e ritagliate in modo tale da avere una dimensione di 512×512 , compatibile con Stable Diffusion.

7.2 Valutazione dei risultati

Non è semplice valutare un modello generativo, verranno quindi mostrate in seguito alcune immagini generate nello stile dei vari artisti così da poter confrontare e valutare visivamente i risultati delle varie metodologie. Tuttavia, è possibile ottenere anche delle valutazioni quantitative grazie a CLIP:

- **Accuracy:** quanto le immagini generate sono coerenti con lo stile dell’artista.
- **Editability:** quanto l’immagine generata sia fedele al prompt testuale fornito in input.

Entrambe le metriche vengono calcolate utilizzando gli encoder per il testo e per le immagini di CLIP, il quale torna molto utile in questo contesto dato che è un modello allenato in modo tale da apprendere embedding di immagini e testi in uno stesso spazio vettoriale.

Dato un artista ed n prompt da valutare:

1. Per ciascun prompt vengono generate 8 immagini nello stile di quell'artista.
2. Le immagini originali e quelle generate vengono codificate utilizzando l'image encoder di CLIP.
3. Viene calcolata la similarità del coseno tra le immagini generate (8) e le immagini originali (4) ottenendo una matrice 4×8 .
4. Si calcola il valore medio tra le 8 immagini generate e poi si prende il valore massimo tra i 4. Questa sarà l'**accuracy**.
5. Codifichiamo il prompt testuale (rimuovendo la parte "... in the style of ...") utilizzando il text encoder di CLIP.
6. Calcoliamo la similarità del coseno tra il prompt e le 8 immagini generate. Il valore medio indicherà la metrica di **editability**.

Questo processo viene ripetuto per ciascun artista nel dataset.

8 Risultati

In questa sezione verranno riportati e commentati i risultati degli esperimenti, sia metriche che immagini valutabili visivamente.

8.1 Risultati complessivi

Nella tabella 1 vengono riportati i valori medi di accuracy ed editability per ciascun approccio presentato nella sezione 6. Osserviamo che la *Textual Inversion* funziona molto meglio rispetto al semplice *Prompt Engineering*, in particolare la variante basata su MoE (**TI-MoE**) ottiene i livelli maggiori di accuracy. Tuttavia possiamo riscontrare un trade-off tra le due metriche: infatti utilizzando un approccio in grado di simulare meglio lo stile di un'artista, i risultati tenderanno ad essere meno fedeli al prompt e a generare meglio i soggetti tipici di quell'artista.

Method	Accuracy	Editability
TI-MoE	0.7584	0.1978
TI-Att	0.7516	0.1968
TI	0.7447	0.2095
PE	0.6797	0.2501

Table 1: Accuracy ed Editability medie per ogni approccio.

8.2 Risultati per artista

Nella tabella 2 vengono invece riportate le metriche raggruppate per artista. Notiamo come per artisti meno famosi (es. Giordano, Mabuse, Pietropoli) l'utilizzo della *Textual Inversion* porta ad ampi miglioramenti rispetto al *Prompt Engineering*, mentre nel caso di artisti più famosi questi miglioramenti sono più ridotti ma pur sempre significativi.

Artist	Accuracy				Editability			
	PE	TI	TI-Att	TI-MoE	PE	TI	TI-Att	TI-MoE
Claude Monet	0.7153	0.7279	0.7535	0.7650	0.2542	0.2284	0.1991	0.2147
Edvard Munch	0.7045	0.6932	0.7443	0.7567	0.2395	0.2113	0.1711	0.1813
Erte	0.6971	0.7666	0.7364	0.7431	0.2312	0.1828	0.1868	0.1976
Eyvind Earle	0.7033	0.7704	0.7559	0.7691	0.2239	0.2000	0.1839	0.1875
Gustav Klimt	0.7012	0.7024	0.7657	0.7639	0.2197	0.2274	0.1906	0.1918
Ilya Mashkov	0.7302	0.7654	0.7653	0.7551	0.2660	0.2614	0.2300	0.2307
Luca Giordano	0.6573	0.7634	0.7451	0.7660	0.2712	0.1661	0.1965	0.1709
Mabuse	0.5783	0.7466	0.7518	0.7092	0.2777	0.1991	0.1971	0.2110
Patrick Pietropoli	0.6158	0.7735	0.7758	0.8192	0.2585	0.1625	0.1723	0.1487
Paul Cezanne	0.6942	0.7375	0.7220	0.7365	0.2586	0.2561	0.2411	0.2435

Table 2: Accuracy ed Editability per ogni artista e metodologia.

8.3 Immagini generate

Dopo aver visto le metriche quantitative, possiamo analizzare visivamente i risultati. Sono state generate quattro immagini per ciascun artista con quattro differenti prompt, in figura 6 possiamo visionare i risultati del *Prompt Engineering*. Notiamo come lo stile di alcuni artisti venga riprodotto abbastanza fedelmente, per esempio *Monet*, *Klimt*, *Giordano*. Per gli altri però lo stile viene riprodotto in maniera completamente errata.

In figura 7 invece osserviamo i risultati ottenuti utilizzando la *Textual Inversion* con *Mixture of Experts*. Notiamo come lo stile di ogni artista venga riprodotto il maniera pressoché perfetta e le immagini generate hanno una qualità superiore al *Prompt Engineering*. L'unica problematica che riscontriamo è la fedeltà al prompt testuale che in alcuni casi viene completamente persa, ad esempio per il “naturalistic landscape” in *Munch*, per entrambi i landscape in *Erte*, per la città in *Giordano* e per i primi tre prompt in *Pietropoli*. In tutti gli altri casi vengono generate delle immagini qualitative e fedeli al prompt, che seguono in maniera accurata lo stile dell'artista.

Parte del problema potrebbe essere dovuto alla poca varianza nelle opere in input, cioè opere che rappresentano sempre lo stesso soggetto. Infatti nel caso di *Patrick Pietropoli* le opere usate nel training sono tutte rappresentazioni di città, mentre nel caso di *Paul Cezanne*, per il quale abbiamo ottimi risultati sia in termini di accuracy che di editability, abbiamo opere rappresentanti quattro soggetti differenti. Bisogna comunque specificare che anche alcuni output ottenuti attraverso il *Prompt Engineering* non seguono correttamente la descrizione fornita in input. Questo potrebbe indicare che la scarsa fedeltà al prompt potrebbe essere riconlegata, in alcuni casi, alla Stable Diffusion stessa.

9 Conclusioni

In questo progetto, abbiamo esplorato diverse metodologie per replicare gli stili distintivi di artisti utilizzando la Stable Diffusion e il metodo di Textual Inversion. Attraverso una serie

di esperimenti, abbiamo valutato l’efficacia di approcci differenti: il Prompt Engineering, la Textual Inversion e varianti di Textual Inversion basate su meccanismi di attenzione e Mixture of Experts (MoE). I risultati ottenuti ci permettono di trarre diverse conclusioni significative. Innanzitutto, l’utilizzo della Textual Inversion si è dimostrato molto efficace nel catturare e replicare lo stile degli artisti rispetto al semplice Prompt Engineering. Questo è particolarmente evidente per artisti meno conosciuti, per i quali il Prompt Engineering non è in grado di ottenere risultati accurati. In particolare, la variante basata su Mixture of Experts (TI-MoE) ha ottenuto i migliori risultati in termini di accuracy, dimostrando che un approccio più sofisticato può migliorare la qualità della simulazione dello stile artistico. Tuttavia, è emerso un trade-off tra l’accuratezza dello stile e la fedeltà al prompt testuale. Infatti, mentre gli approcci di Textual Inversion migliorano la capacità del modello di replicare lo stile dell’artista, tendono a ridurre la fedeltà all’input testuale, evidenziando una minore editability. Questo fenomeno suggerisce che, mentre si ottimizza per uno stile più accurato, la rappresentazione visiva tende a divergere leggermente dal contenuto esatto del prompt. Future ricerche potrebbero concentrarsi su ulteriori miglioramenti di queste metodologie, ad esempio integrando tecniche per ridurre il trade-off tra accuracy ed editability.

10 Sviluppi Futuri

A causa di limiti computazionali, la Textual Inversion è stata allenata solo per 1000 step ad artista, probabilmente incrementando il numero di iterazioni potremmo raggiungere dei risultati migliori. Inoltre abbiamo visto come utilizzando modelli più complessi siamo in grado di catturare e replicare meglio lo stile di un’artista, tuttavia la capacità di rimanere fedeli al prompt testuale diminuisce rispetto ad approcci più semplici. Le immagini generate secondo lo stile catturato dai modelli più avanzati tendono a replicare molto bene i soggetti tipici di quell’artista, mentre sono poco in grado di generalizzare e generare immagini rappresentanti soggetti completamente diversi. Un possibile sviluppo futuro potrebbe essere quello di risolvere questa problematica di “*overfitting*” sulle opere degli artisti, per esempio aumentando il numero di opere per artista, incrementando la varianza delle opere (soggetti diversi) o utilizzando prompt testuali più specifici per ciascuna opera in input durante il training, ad esempio utilizzando l’output di un captioner che descriva l’opera, piuttosto che utilizzare prompt generici come fatto durante questi esperimenti.

References

- [1] Giovanna Castellano et al. “Leveraging Knowledge Graphs and Deep Learning for automatic art analysis”. In: *Know.-Based Syst.* 248.C (July 2022). ISSN: 0950-7051. DOI: 10.1016/j.knosys.2022.108859. URL: <https://doi.org/10.1016/j.knosys.2022.108859>.
- [2] Rinon Gal et al. *An Image is Worth One Word: Personalizing Text-to-Image Generation using Textual Inversion*. 2022. arXiv: 2208.01618 [cs.CV].
- [3] Alec Radford et al. *Learning Transferable Visual Models From Natural Language Supervision*. 2021. arXiv: 2103.00020 [cs.CV].
- [4] Carlos Riquelme et al. *Scaling Vision with Sparse Mixture of Experts*. 2021. arXiv: 2106.05974 [cs.CV].

- [5] Robin Rombach et al. *High-Resolution Image Synthesis with Latent Diffusion Models*. 2022. arXiv: 2112.10752 [cs.CV].
- [6] Ashish Vaswani et al. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL].
- [7] Yuxin Zhang et al. *Inversion-Based Style Transfer with Diffusion Models*. 2023. arXiv: 2211.13203 [cs.CV].

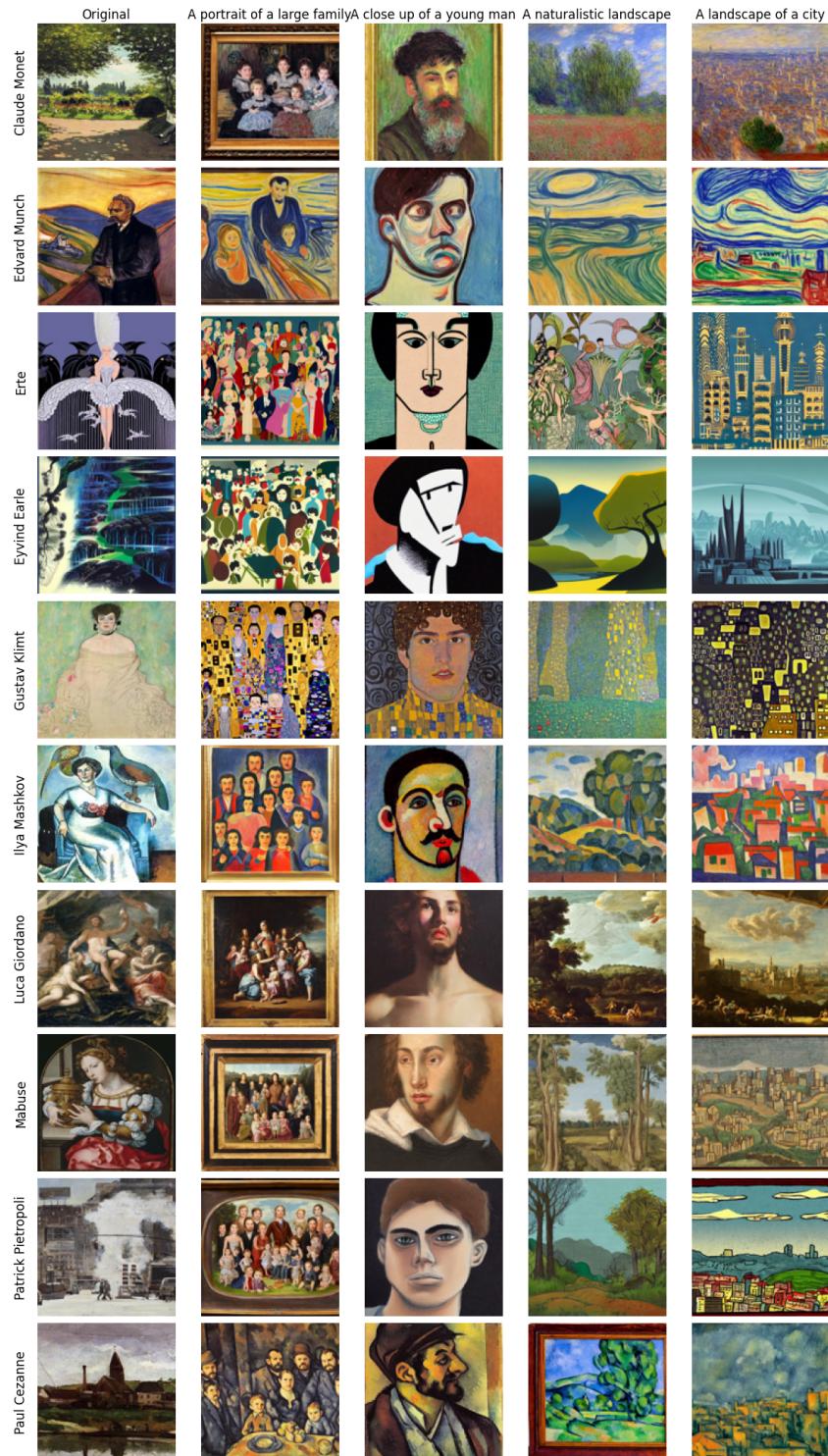


Figure 6: Immagini generate con il Prompt Engineering.

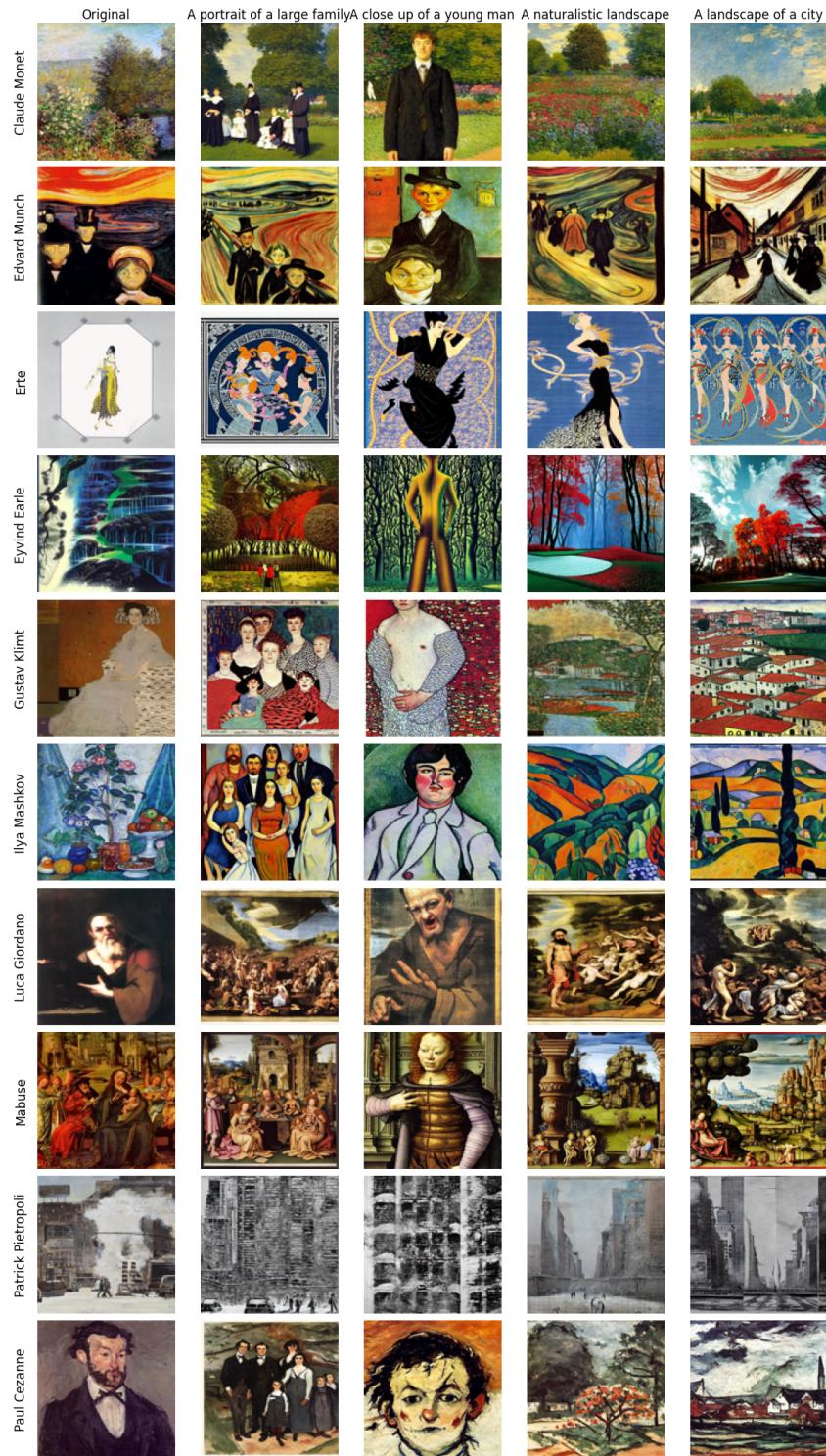


Figure 7: Immagini generate con la Textual Inversion (MoE).