# Sperimentazione nell'allenamento di modelli encoder-decoder per l'image captioning su WikiArt

Amorosini Andrea
Universita' degli Studi di Salerno, Dipartimento di Informatica

***Abstract*— Questo Paper mira ad esporre la sperimentazione effettuata con vari modelli in architettura encoder-decoder per un task di generazione delle didascalie sulle immagini, nello specifico sperimentando le performance di tali modelli nel caso di un allenamento esclusivamente su opere d'arte e valutandoli sfruttando le metriche BLEU, ROUGE e METEOR, così come un test finale ad occhio umano.**
**Dai risultati finali sembrerebbe che la scelta finale di un modello Swin2-GPT2 allenato sul dataset WikiArt riporti una maggiore precisione nelle didascalie generate andando a surclassare modelli simili come ViT-GPT2.**

***Index Terms*— Encoder-Decoder, Captioning, VisionEncoderDecoderModel, WikiArt**

## I. INTRODUZIONE

La generazione di didascalie per immagini e' un problema analizzato negli anni e per cui sono stati sfruttati diversi approcci. Si e' partiti dalle implementazioni facenti uso di reti convoluzionali (CNN) e ricorrenti (RNN), continuando con le ultime implementazioni che sfruttano esclusivamente i transformers per la generazione di didascalie e finendo cosi' con l'integrazione di questo task all'interno di grossi modelli multimodali.

L'obiettivo di questa ricerca consiste nella sperimentazione di alcune implementazioni di transformers allo scopo di creare modelli che vadano esclusivamente a generare didascalie per dipinti di vario genere e stile.

Nello specifico si vuole analizzare come le didascalie generate possano cambiare rispetto allo stato dell'arte allenando il modello esclusivamente su dipinti piuttosto che allenandoli con dataset generici come ImageNet o Flickr in una delle loro svariate varianti.

In questa ricerca ci si focalizza su una implementazione Encoder-Decoder usando transformers sia come image encoder che come text decoder e cercando di estrarre didascalie esclusivamente oggettive, ovvero senza ricorrere a nessuna informazione data dal contesto dietro all'opera.

Questo paper e' strutturato come segue: si inizia con un analisi sullo stato dell'arte per la generazione di didascalie, continuando con una breve descrizione dei materiali e metodi utilizzati lungo il corso di questa ricerca, per poi passare ad una descrizione dell'ambiente di sperimentazione andando poi a finire con un report dei risultati ottenuti.

## II. STATO DELL'ARTE

Nel campo della generazione di didascalie per immagini ci sono stati notevoli progressi in anni recenti grazie all'avvento di tecniche avanzate di deep learning e all'introduzione dell' IA generativa. Attualmente questi modelli si basano principalmente su architetture encoder-decoder[2], dove una rete convoluzionale (CNN) o un Vision Transformer funge da encoder per estrarre le caratteristiche visive dall'immagine, e una rete ricorrente (RNN) o un Transformer che funziona da decoder per generare la didascalia. Modelli come il "Show, Attend, and Tell"[24] hanno portato all'introduzione di tecniche avanzate come l'uso di meccanismi di attenzione, permettendo al modello di concentrarsi su piu' parti dell'immagine e generando cosi' didascalie maggiormente rilevanti e coerenti.

Piu' di recente, architetture come il Vision Transformer (ViT)[5] e i modelli multimodali come CLIP[20] e DALL-E[22] da OpenAI hanno fatto progredire ancor di piu' lo stato dell'arte, grazie all' uso di tecniche di autoattenzione per catturare relazioni complesse tra le caratteristiche dell'immagine e quelle del testo. Inoltre, l'uso di grossi dataset annotati, tecniche di pre-addestramento su dati non supervisionati e l'integrazione di modelli linguistici di grandi dimensioni (come GPT3/4 o Claude) hanno permesso di ottenere descrizioni sempre piu' accurate e naturali.

Nonostante tutti questi progressi, rimangono comunque altre sfide come la gestione di immagini più complesse o ambigue, la generazione di didascalie creative e personalizzate, e l'assicurazione di equità e riduzione dei bias nei modelli.

La ricerca continua, quindi, a esplorare nuove architetture, tecniche di allenamento, dataset e metodologie per affrontare queste sfide rimanenti e aumentare ulteriormente la qualita' delle didascalie generate.

## III. METODOLOGIE E STRUMENTI

### A. Dataset

È stato scelto come Dataset WikiArt[9], un ampio dataset di immagini artistiche, estrapolato dall'enciclopedia artistica online[23] con lo stesso nome, contenente piu' di 80000 (delle oltre 250000 online) opere d'arte provenienti da una vasta gamma di periodi storici, stili e artisti, includendo pitture rinascimentali, impressioniste, moderne per arrivare fino alle contemporanee. Ogni immagine nel dataset originale è accompagnata da metadata dettagliati, come l'artista, il titolo, l'anno di creazione, il genere e lo stile dell'opera.

*1) PreProcessing:* Per poter adattare questo dataset al task di generazione di didascalie lo stesso è stato sottoposto ad un operazione di generazione di didascalie effettuata con un modello multimodale Open-Source "MoonDream2"[17] disponibile liberamente sull'hub di huggingface. Sono state prima testate le capacità del modello con svariati sample casuali dal dataset ed una volta confermate le performance di generazione soddisfacenti è stato dato al modello lo stesso prompt "Genera una didascalia breve, semplice e solo visivamente descrittiva per questa immagine." per ogni immagine nel dataset.

Una volta finita l'operazione di generazione si e' passati ad un controllo sui dati generati, per quanto possibile, controllando prima la lunghezza media delle didascalie generate ed isolando eventuali outlier, che si sono scoperti avere didascalie insolitamente lunghe per colpa di un errore di generazione del modello utilizzato in presenza di scritte stilizzate all'interno delle opere.

Tutte queste operazioni cosi come il caricamento iniziale del dataset sono stati eseguiti usando la libreria Datasets di Huggingface sfruttando i metodi di mapping e filtraggio nativi della libreria.

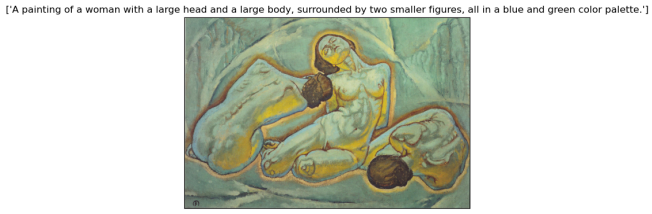


Fig. 1. Esempio di opera estratta dal Dataset con la nuova didascalia generata

Infine e' stato applicato un train-validation-test split con un training set corrispondente all'80% del dataset, un validation set corrispondente all'10% del dataset e il test set corrispondente al 10% del dataset.

Inoltre e' stato testato l'allenamento sia con l'intero dataset che con solo la sua meta' per provare a velocizzare la fase di training osservando se le performance risultassero essere simili.

### *B. Architettura*

E' stato deciso in fase preliminare di concentrarsi su un architettura Encoder-Decoder.

Questa architettura viene ampiamente utilizzata per compiti di apprendimento sequenziale e traduzione, come la traduzione automatica, il riconoscimento vocale e il captioning di immagini. E' composta da due parti principali: l'Encoder e il Decoder. L'Encoder riceve una sequenza di input (ad esempio, parole di una frase o caratteristiche estratte da un'immagine) e la trasforma in una rappresentazione interna, spesso chiamata vettore di contesto o embedding. Per le immagini, l'Encoder può essere una rete convoluzionale (CNN) o anche un modello Vision Transformer. Mentre il Decoder riceve il vettore di contesto prodotto dall'Encoder e genera una sequenza di

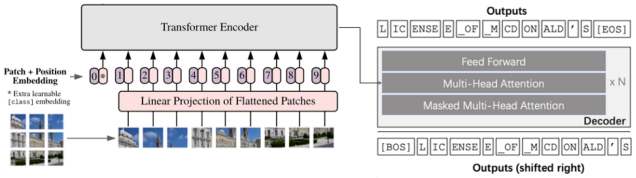


Fig. 2. Architettura VisionEncoderDecoder

output, elemento per elemento. Il Decoder può essere una RNN, LSTM o GRU per i dati sequenziali, o una rete specifica per la natura del compito. Il processo di decodifica è spesso condizionato non solo dal vettore di contesto ma anche dai passi di decodifica precedenti, il che permette alla rete di generare output coerenti e significativi.

Nel corso di questa di ricerca si e' deciso di focalizzarsi sull'uso di Vision Transformers come encoder e Transformers come decoder così come illustrato nella Figura2.

Per fare cio' e' stata utilizzata l'architettura VisionEncoderDecoderModel[11] dalla libreria Transformers di Huggingface cosi' da poter sperimentare con diverse combinazioni di modelli velocemente ed evitando grossi cambiamenti alla logica di allenamento. Viene illustrata la sua struttura nella figura 2.

### *C. Metriche di Valutazione impiegate*

Per la fase di valutazione finale sono stati impiegate le metriche BLEU[19], ROUGE[13] e METEOR[1] dalla libreria evaluate di Huggingface per poter valutare le didascalie generate rispetto alle didascalie contenute nel dataset.

*1) BLEU:* BLEU è una metrica per valutare la qualità del testo generato da un modello confrontandolo con uno o più testi di riferimento umani. Viene calcolata confrontando le n-gramme (sequenze di parole contigue) del testo generato con quelle del testo di riferimento e misurando la precisione. Più le n-gramme del testo generato corrispondono a quelle del testo di riferimento, più alto sarà il punteggio BLEU, che varia da 0 a 1.

*2) ROUGE:* ROUGE è un insieme di metriche utilizzate per valutare la qualità dei testi generati automaticamente, confrontandoli con testi di riferimento umani. Le varianti più comuni includono:

- ROUGE-N: Calcola il recall basato sugli n-gramme.
- ROUGE-L: Misura la similarità basata sulle più lunghe sottosequenze comuni (LCS).
- ROUGE-S: Basato sulle coppie di skip-bigram (coppie di parole che possono essere non contigue).

*3) METEOR:* METEOR è una metrica di valutazione della generazione del testo che migliora BLEU tenendo conto di corrispondenze di sinonimi, stemming (riduzione delle parole alla loro radice) e allineamento degli ordini di parole. METEOR combina precisione, recall e una penalizzazione per allineamenti discontinui per fornire una

valutazione più dettagliata. I punteggi METEOR variano da 0 a 1, con punteggi più alti che indicano una migliore qualità della traduzione.

### D. Librerie di utility

Inoltre, per poter tenere in maniera ordinata tutti i dati di ogni singolo esperimento di allenamento e' stata poi integrata la piattaforma ClearML[3] per poter registrare in maniera efficiente tutti i parametri usati nell'esperimento, l'andamento della loss di allenamento e validazione e le metriche finali risultanti.

Ed infine per poter esplorare e visualizzare il dataset sono state impiegate le librerie MatPlotLib e Pandas.

## IV. VALUTAZIONE SPERIMENTALE

### A. Scelta dei modelli

Visti i lunghi tempi di allenamento che questa architettura presentava si e' deciso fin da subito di restringere il campo dei modelli da prendere in considerazione come encoders e decoders.

*1) Encoders:* Dopo alcune prove iniziali con vari encoder come ViT-Base, Beit e Deit ci si e' infine focalizzati sul modello Swin2[16] di Microsoft. Tale scelta grazie ad alcune ricerche che hanno portato alla luce come Swin 2 potrebbe restituire caratteristiche maggiormente discriminanti da opere d'arte grazie alla sua caratteristica distintiva della Finestra di Spostamento[15] (Shifted Windows), la quale aiuta a raffinare le capacita' del modello di catturare le dipendenze a lungo raggio tra le patch dell'immagine mantenendo alte prestazioni e allo stesso tempo dimostrandosi più efficiente di altri modelli simili.

*2) Decoders:* Per quanto riguarda il decoder invece, e' stato scelto un pool iniziale di modelli tra i piu' utilizzati in letteratura e dalla community:

- BERT[8][4] : Modello per la generazione di sequenza a sequenza sviluppato da Google e pre-addestrato su grandi corpi di testo utilizzando Masked Language Modeling (MLM) e Next Sentence Prediction (NSP).
- BART[6][12] : Modello di generazione sequenza a sequenza sviluppato da Facebook AI che combina le caratteristiche di BERT e GPT. Addestrato in due fasi, denoising autoencoder e generazione di testo.
- ROBERTA[7][14] : Variante di BERT sviluppata da Facebook AI, ottimizzata per una maggiore efficienza e performance. Le principali differenze includono un addestramento su dati più grandi, rimozione del compito di NSP e utilizzo di batch più grandi e un periodo di addestramento più lungo.
- GPT2[18][21] : Modello di linguaggio autoregressivo sviluppato da OpenAI, a differenza di BERT, GPT-2 è unidirezionale e predice la parola successiva in una sequenza di testo, addestrato su un ampio corpus di dati web. È noto per la sua capacità di generare testi coerenti e realistici, completare frasi e svolgere vari compiti di NLP con pochissimo fine-tuning.

Dopo svariati test e' pero' risultato che BERT, BART e ROBERTA non fossero in grado di generare testo sensato dalle feature fornite da Swin2 o da nessuno degli altri encoder provati. Si e' deciso cosi' di focalizzare gli sforzi sull'utilizzo di GPT2 che risultava essere l'unico a generare testo rilevante a cio' che veniva presentato nell'opera.

### B. Generazione e impostazione iniziale del modello finale

Per ogni combinazione provata grazie all'utilizzo di VisionEncoderDecoderModel le impostazioni iniziali erano comuni a tutte, generando le Configurazioni di ogni modello iniziale ed eventuali utility come l'image processor corrispondente per l'image encoder o il tokenizer corrispondente per il text decoder e abilitando la cross-attention sul decoder.

Una volta generato il VisionEncoderDecoderModel, passandogli i modelli scelti e le configurazioni generate, è necessario impostare un decoder_start_token_id e un pad_token_id.

Per tutti i modelli è stato scelto come decoder_start_token_id il cls_token_id o il bos_token_id a seconda di quale fosse presente nei tokenizer corrispondenti, mentre per il pad_token_id l'unico caso particolare era quello di GPT2 che non presenta un PAD token di default, e al quale è stato quindi aggiunto manualmente come ([PAD]).

### C. Impostazioni di generazione

Tra i parametri testati troviamo:

- num_beams: per aumentare i dettagli catturati dal modello nell'opera,
- no_repeat_ngram_size : per evitare problemi di ripetizione di parole nella didascalia generata,
- num_beam_groups : per provare a differenziare ulteriormente le informazioni ottenute dall'opera suddividendo i beams specificati in num_beams in piu' gruppi,
- diversity_penalty : per impostare il grado di diversità desiderato da ognuno dei gruppi di beam tra di loro.

### D. Classi del Dataset e Dataloader

Una volta impostato completamente il modello è stata descritta una classe nativa Pytorch per il dataset che includesse le operazioni di preprocessing sulle singole immagini così come descritte all'interno dell'image encoder scelto e la tokenizzazione della didascalia contenuta nel dataset con il tokenizer corrispondente al decoder scelto in maniera tale che queste venisserro elaborate solo a tempo di esecuzione.

Questa classe e' stata poi utilizzata per la creazione dei data loaders corrispondenti per ogni split con una dimensione di batch impostata su 8 per mancanza di RAM dedicata alla GPU che non permetteva di sperimentare con dimensioni maggiori date le dimensioni del dataset e del modello da allenare.

### E. Funzioni e parametri di allenamento

E' stato poi creata una funzione di training in PyTorch nativo con fasi di training e validazione per ogni epoca includendo la possibilita' di uso di uno scheduler per il learning rate, e implementando una tecnica di gradient accumulation per far fronte all'impossibilita' di aumentare la dimensione dei batch.

*1) Loss e optimizer:* Come funzione di loss e' stata usata la Cross Entropy cosi' come fornita direttamente dal VisionEncoderDecoderModel, mentre come optimizer si e' scelto AdamW con un learning rate variabile da 5e-03 a 7e-05 e con l'aggiunta di una weight decay da 1e-05 a 1e-06 dopo i primi allenamenti che mostravano come il modello andasse in overfitting.

*2) Scheduler:* Mentre come scheduler si e' scelto di usare il cosine_schedule_with_warmup fornito da huggingface che imposta una fase di warmup, impostata sul 20% delle epoche di allenamento, dove il learning rate aumenta lentamente da 0.0 fino al learning rate specificato all'optimizer cosi' da poter analizzare durante l'allenamento se fosse necessario aumentare o diminuire il learning rate impostato.

### F. Valutazione finale

Infine, una volta finito l'allenamento, vengono fatte generare le didascalie per tutto il test set per poi passarle assieme alle didascalie contenute nel dataset alle funzioni per calcolare lo score BLEU, ROUGE e METEOR per poter valutare la somiglianza con le didascalie originali il che, pero', potrebbe non catturare completamente se il modello stia effettivamente comprendendo o no cio' che gli si presenta. Per questo oltre alle metriche sopra elencate sono state selezionate una serie di immagini da vari periodi artistici per poter valutare con occhio umano le didascalie generate e poter confrontare le performance con altri modelli di generazione delle didascalie. I periodi artistici selezionati con i loro dipinti corrispondenti includono:

- Rinascimento : "L'ultima Cena, Leonardo Da Vinci, 1495-1498"
- Barocco : "Vocazione di San Matteo, Caravaggio, 1599-1600"
- Rococo : "L'Altalena, Jean-Honorè Fragonard, 1767"
- Neoclassicismo : Il Giuramento degli Orazi, Jacques-Louis David, 1784"
- Romanticismo : "3 Maggio 1808, Francisco Goya, 1914"
- Realismo : "Les Casseurs de pierres, Gustave Courbet, 1849"
- Impressionismo : "Notte stellata, Vincent Van Gogh, 1889"
- Simbolismo : "Le cyclope, Odilon Redon, 1840-1916"
- Art Nouveau : "Il Bacio, Gustav Klimt, 1907-08"
- Favismo : "Donna con cappello, Henri Matisse, 1905"
- Espressionismo : "L'Urlo, Edvard Munch, 1893"
- Cubismo : "Les Demoiselles d'Avignon, Pablo Picasso, 1906-07"
- Futurismo : "Cyclist, Natalia Goncharova, 1913"
- Dada : "The Art Critic, Raoul Hausmann, 1919-20"
- Surrealismo : "La Persistenza della Memoria, Salvador Dalì, 1931"
- Espressionismo Astratto : "No. 5 1984, Jackson Pollock, 1984"
- Pop Art : "Serie De 10 Marilyn Monroe, Andy Warhol, 1967"
- Minimalismo : "Composition en rouge, jaune, bleu et noir, Piet Mondriaan, 1921"
- Arte Contemporanea : "Shop Until You Drop, Banksy, 2005"

## V. PRESENTAZIONE DEI RISULTATI OTTENUTI

Nel corso della sperimentazione è stato isolato il modello con le performance migliori sia come metriche specificate precedentemente sia giudicando in base ai risultati sulle immagini di test selezionate sopra riportate.

Come configurazione dei modelli di base, come sopra già detto, alla fine ci si è focalizzati su un modello Swin2-GPT2, allenato solo con metà del dataset selezionato con i seguenti parametri:

| | |
|---|---|
| num_beams | 8 |
| no_repeat_ngram_size | 0.1 |
| num_beam_groups | 1 |
| diversity_penalty | 0.0 |

TABLE I

PARAMETRI PER LA GENERAZIONE

| | |
|---|---|
| num_epochs | 15 |
| learning_rate | 7e-05 |
| batch_size | 8 |
| weight_decay | 1e-06 |
| accumulation_steps | 8 |

TABLE II

PARAMETRI PER L'ALLENAMENTO

Da questo allenamento si è potuto osservare un, quasi, stabile andamento decrescente nella funzione di loss, mostrando anche una stretta correlazione tra la loss di allenamento e validazione.

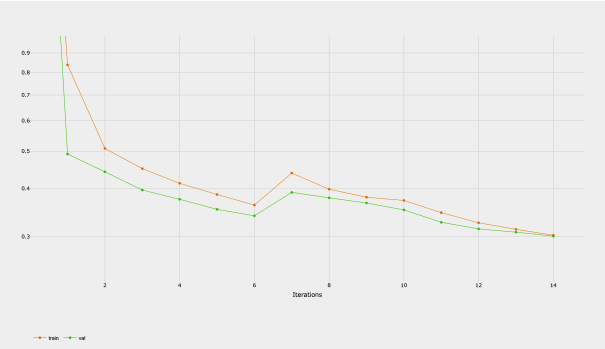


Fig. 3. Andamento della funzione di loss per Allenamento (Gialla) e Validazione (Verde)

- Training-Loss Finale : 0.3025601
- Validation-Loss Finale : 0.3009027

Per quanto riguarda le metriche calcolate alla fine dell'allenamento, dal modello scelto sono risultate le seguenti metriche:

| | |
|---|---|
| ROUGE1 | 0.5727 |
| ROUGE2 | 0.3645 |
| ROUGEL | 0.5145 |
| ROUGELSum | 0.5143 |

TABLE III
ROUGE SCORES

| | |
|---|---|
| BLEU | 0.29 |
| Precision-1 | 0.5795 |
| Precision-2 | 0.3345 |
| Precision-3 | 0.2264 |
| Precision-4 | 0.1611 |

TABLE IV
BLEU SCORES

| | |
|---|---|
| METEOR | 0.5010 |

TABLE V
METEOR SCORE

Si può quindi vedere come, in base sopratutto agli score ROUGEL, ROUGE1 e METEOR che il modello riesce a ricreare almeno metà della didascalia originale del dataset.

Però, come già detto precedentemente, l'uso di queste metriche non basta a valuatare le performance di questo task, passo quindi a presentare dei confronti su alcune delle immagini di test prima elencate tra il modello sviluppato ed uno dei modelli piu' utilizzati ed anche allo stesso tempo il piu' vicino all'architettura presentata qui, ovvero ViT-GPT2[10].

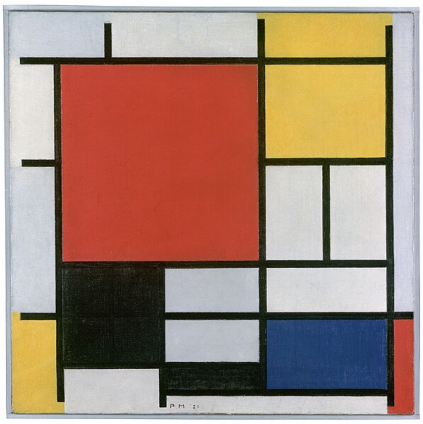

Fig. 4. Composition en rouge, jaune, bleu et noir, Minimalismo

- Vit-GPT2 : "a colorful wall with a red and white stripe"
- Swin2-GPT2 : "A square with a grid of squares in red, black, and blue."

Fig. 5. Les Demoiselles d'Avignon, Cubismo

- Vit-GPT2 : "a painting of a woman in a bikini"
- Swin2-GPT2 : "A painting of four naked women in a room with a blue and orange background."

Fig. 6. La Persistenza della Memoria, Surrealismo

- Vit-GPT2 : "a woman is sitting on a beach with a surfboard"
- Swin2-GPT2 : "A painting of a man in a white shirt and brown pants lying on a beach, with a blue sky and ocean in the background."

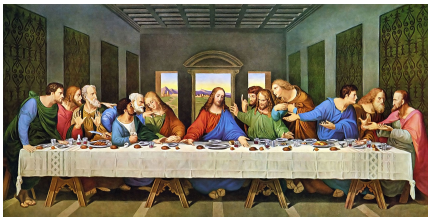

Fig. 7. L'ultima Cena, Rinascimento

- Vit-GPT2 : "a large group of people sitting around a table"
- Swin2-GPT2 : "A painting of Jesus Christ surrounded by his twelve Apostles, with a chandelier above and a window on the left side."

Fig. 8. No. 5 1984, Espressionismo Astratto

- Vit-GPT2 : "a pile of dirt with a few leaves"
- Swin2-GPT2 : "A painting with a chaotic background of black, yellow, and red splatters and lines."

Fig. 9. El Tres de Mayo, Romanticismo

- Vit-GPT2 : "a painting of a man in a black and white outfit"
- Swin2-GPT2 : "A group of people in military uniforms stand around a table, with one person lying on the ground and another holding a gun."

Fig. 10. Il bacio, Art Noveau

- Vit-GPT2 : "a painting of a person with a blue shirt and a blue scarf"
- Swin2-GPT2 : "A painting of a woman in a yellow and blue dress with red flowers, standing in front of a green wall."

Possiamo quindi osservare come rispetto a ViT-GPT2, allenato su un dataset di foto che riproducono la realtà, Swin2-GPT2 riesce correttamente a riconoscere anche soggetti "astratti" e ad andare molto più nello specifico nella descrizione dell'opera, condividendo però alcune lacune; come si può vedere nell'esempio con l'opera Surrealista dove entrambi i modelli hanno fallito nel riconoscere anche solo un minimo la scena che avevano di fronte condividendo, però, il riconoscimento di una spiaggia all'interno dell'opera. Inoltre nel modello allenato si possono comunque ancora notare lievi allucinazioni come nel caso della figura 7 dove il modello identifica un lampadario dove non ce ne è nessuno. Questi sono solo alcuni degli esempi con cui è stato sperimentato il modello, ma generalmente si può dire che il modello qui presentato abbia ricevuto dei grossi vantaggi nell'essere

stato allenato anche solo con metà del dataset WikiArt e nonostante alcune limitazioni computazionali che non hanno permesso la piena esplorazione dello spazio dei parametri.

## VI. CONCLUSIONI

Dalle sperimentazioni fatte, si e' mostrata la robustezza di Swin2 nell'estrazione di caratteristiche pertinenti rispetto a modelli come ViT, ma sicuramente a questo task gioverebbe avere una maggiore capacita' computazionale disponibile e la possibilita' di eseguire esperimenti in parallelo su piu' combinazioni di modelli, includendo anche le varianti dei modelli presentati in questo paper.

Inoltre sarebbe utile provare ad introdurre informazioni esterne sul contesto dell'opera in esame per vedere se il modello riesca a sfruttare queste informazioni per arricchire la descrizione dell'opera.

## REFERENCES


[1] Satanjeev Banerjee and Alon Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In Jade Goldstein, Alon Lavie, Chin-Yew Lin, and Clare Voss, editors, *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.

[2] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation, 2014.

[3] ClearML. allegroai/clearml, Jul 2024.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

[6] Facebook. facebook/bart-base, Jul 2024.

[7] FacebookAI. Facebookai/roberta-base, Jul 2024.

[8] Google. google-bert/bert-base-uncased.

[9] Huggingface. huggan/wikiart-datasets at hugging face, Jul 2024.

[10] Huggingface. nlpconnect/vit-gpt2-image-captioning, Jul 2024.

[11] Huggingface. Vision encoder decoder models, Jul 2024.

[12] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, 2019.

[13] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[14] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019.

[15] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, Furu Wei, and Baining Guo. Swin transformer v2: Scaling up capacity and resolution, 2022.

[16] Microsoft. microsoft/swinv2-base-patch4-window12to16-192to256-22kto1k-ft, Jul 2024.

[17] MoonDream. vikhyatk/moondream2, Jul 2024.

[18] OpenAI. openai-community/gpt2, Jul 2024.

[19] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In Pierre Isabelle, Eugene Charniak, and Dekang Lin, editors, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics.

[20] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

[21] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[22] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation, 2021.

[23] WikiArt.

[24] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention, 2016.