Deep Learning Project:

Image Captioning on the COCO Dataset

University of Milano-Bicocca

UNIVERSITA' DEGLI STUDI DI MILANO BICOCCA

Our team

Alex Calabrese

Data Science Student

Antonio Sabbatella

Data Science Student

Table of contents

COCO (Common Objects in Context)

- Large-scale object detection, segmentation, and **captioning** dataset 2014.
- Contains over **330,000 images** with annotated objects.
- Widely used in computer vision research and development.
- **<u>Size: 17 GB.</u>**

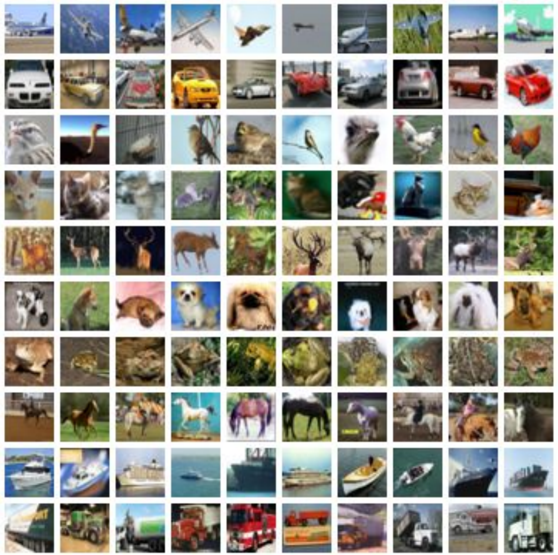


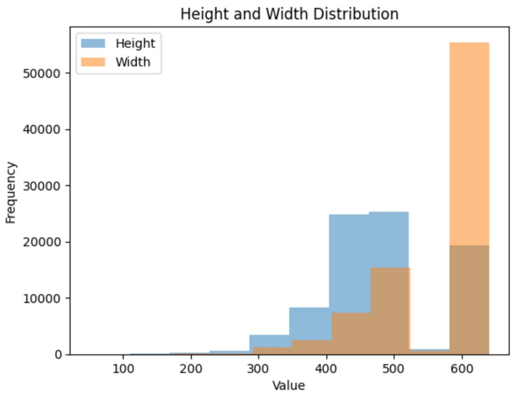
Image Captioning

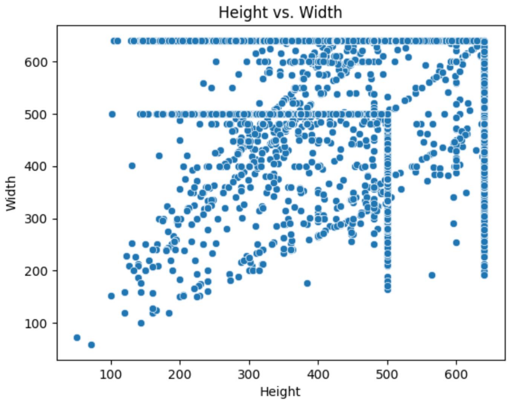
Image captioning is the process of generating a textual description for given images.

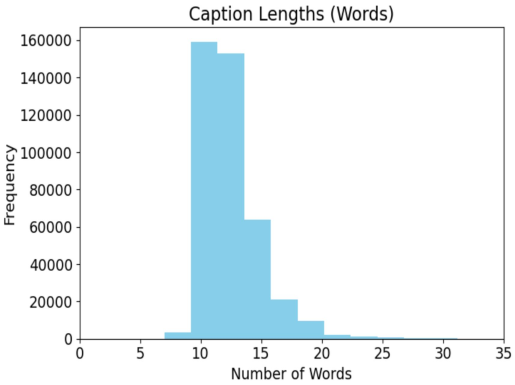
Input:

Output: *"A cat lying on a couch with a remote lying next to it."*

Exploration





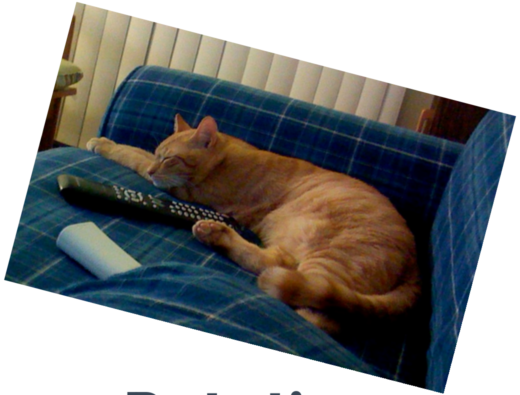


Heights are clustered **~500 pixels** while **widths** near **~600 pixels**.
Captions contain between **10 to 25 words** with a peak around 12 words.

Base data augmentation

Flip

Randomly flips image horizontally
Factor: 50% chance



Rotation

Rotates image by r. angle
Factor: ±0.2 radians (±11.50)

Contrast

Alters contrast levels by a factor ±0.3

Experiments with Custom Models

LSTM

LSTMs are recurrent neural networks that handle long-term dependencies in sequential data using memory cells and gates.

Transformer

Transformers, use self-attention mechanisms to process entire sequences in parallel, capturing global dependencies enabling better performance on many language tasks.

Loss Functions

Kullback–Leibler divergence

Measures the **dissimilarity between** two **probability distributions**

$$\frac{\Sigma_x P(x) log \frac{P(x)}{Q(x)}}{n}$$

Categorical Focal Cross entropy

Focuses on **difficult-to-classify** examples by **down-weighting** the loss for **well-classified instances**

$$-\alpha_t (1 - p_t)^\gamma \log(p_t)$$

Sparse Categorical Cross Entropy

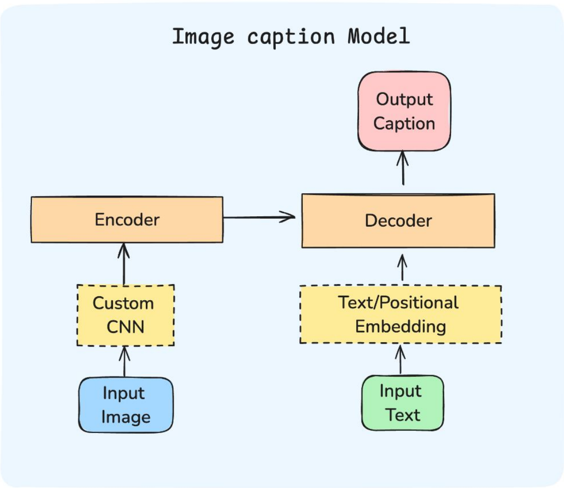
Uses integer-encoded labels instead of one-hot encoding, efficiently **penalizing predictions** based on their **deviation** from the **true class** label

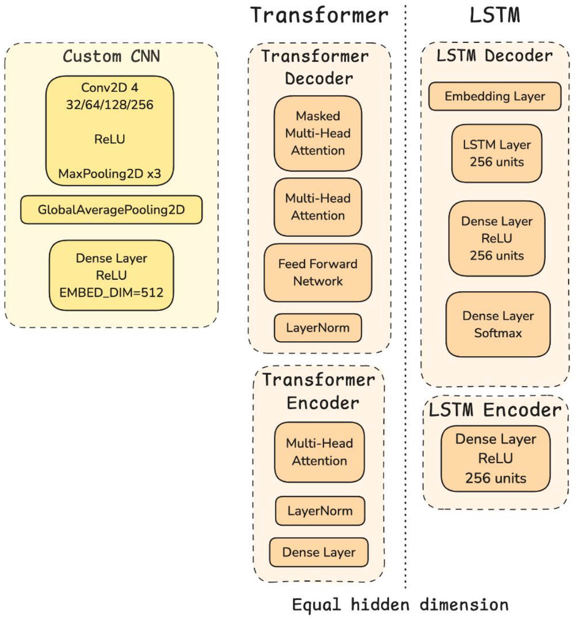
$$-\sum_{i=1}^{n} t_i \log(p_i),$$

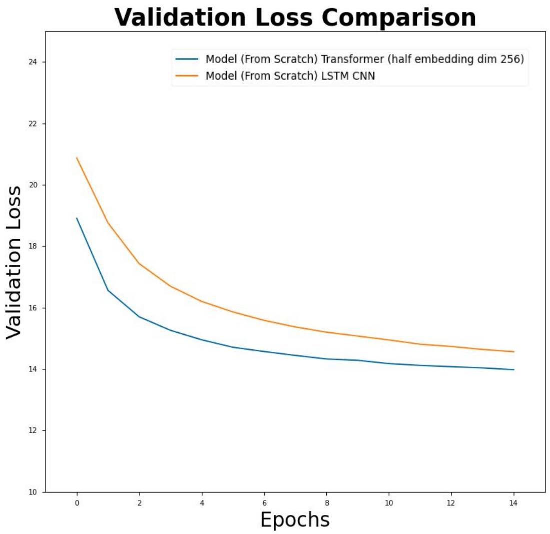
t trouth label,
p softmax probability

The tests in the following slides use the Sparse Categorical Cross Entropy.

LSTM vs Transformer

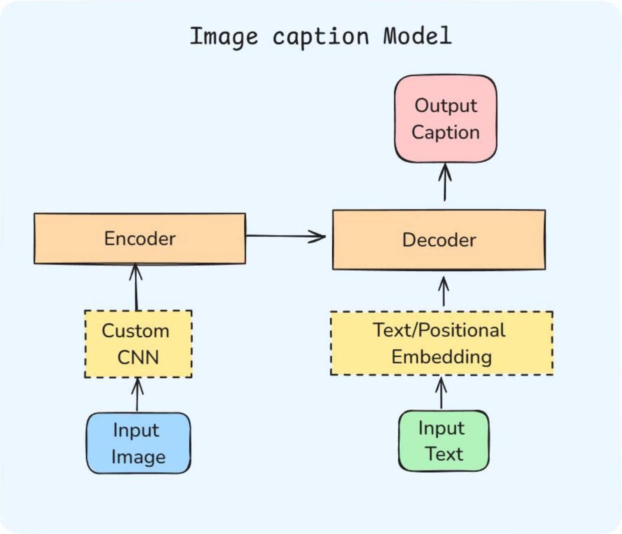


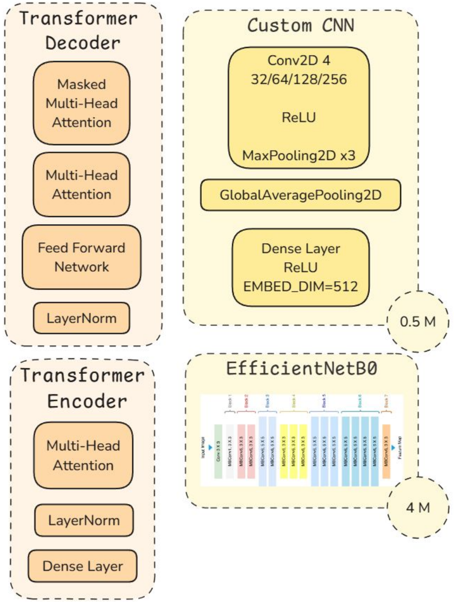


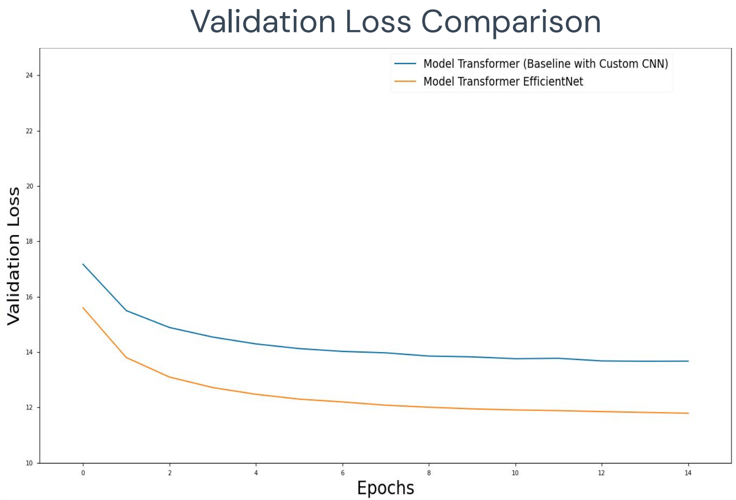


Model based on **transformers outperforms** the model based on LSTM.

Custom CNN vs EfficientNet





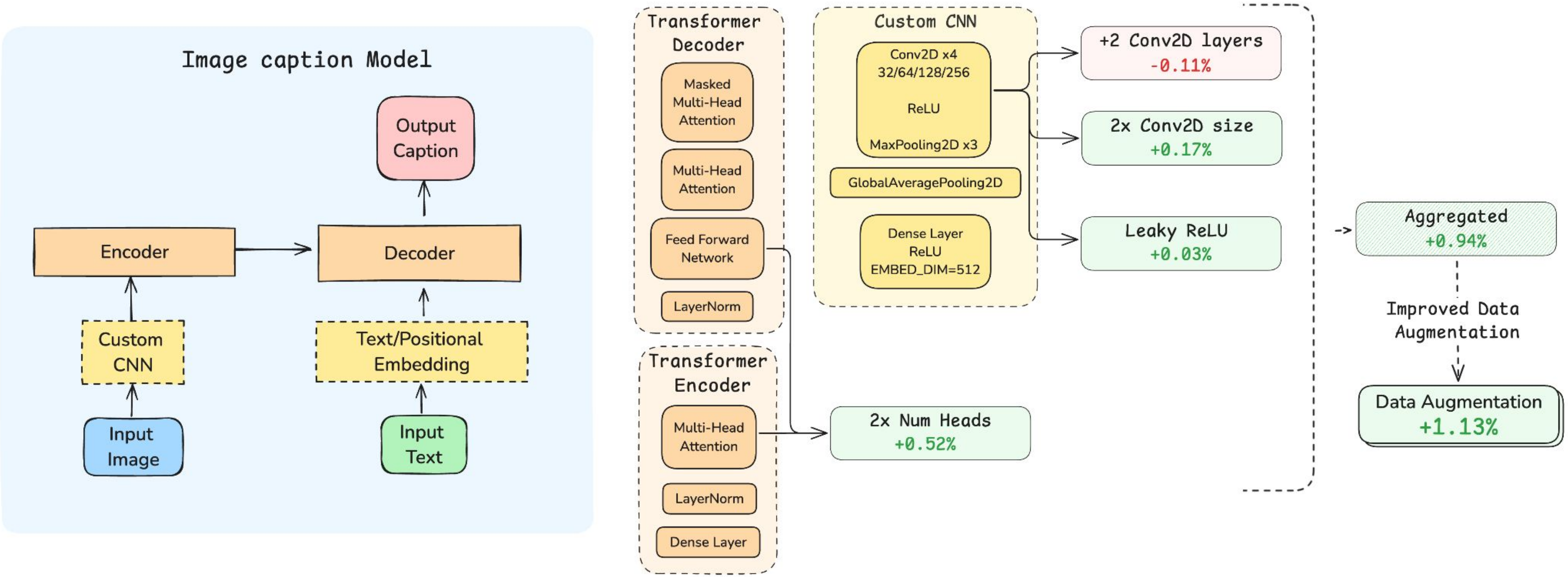


Pre-trained model based on **EfficientNet outperforms** the Model with a Custom CNN.

Pretrained (EfficientNetB0) > Custom CNN

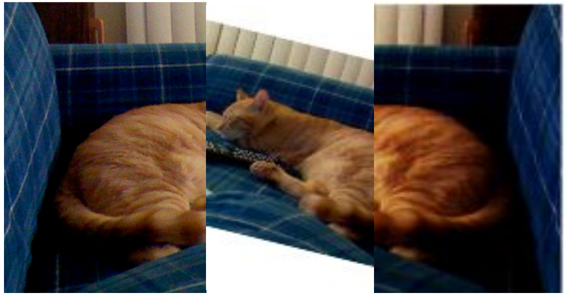
Let's see if we can match EfficientNet!

Architecture Optimization



+1.13% Improvement compared to Base Transformer with Custom CNN model

Improved data augmentation



Base

Random Horizontal flip,
Random Rotation,
Random Contrast

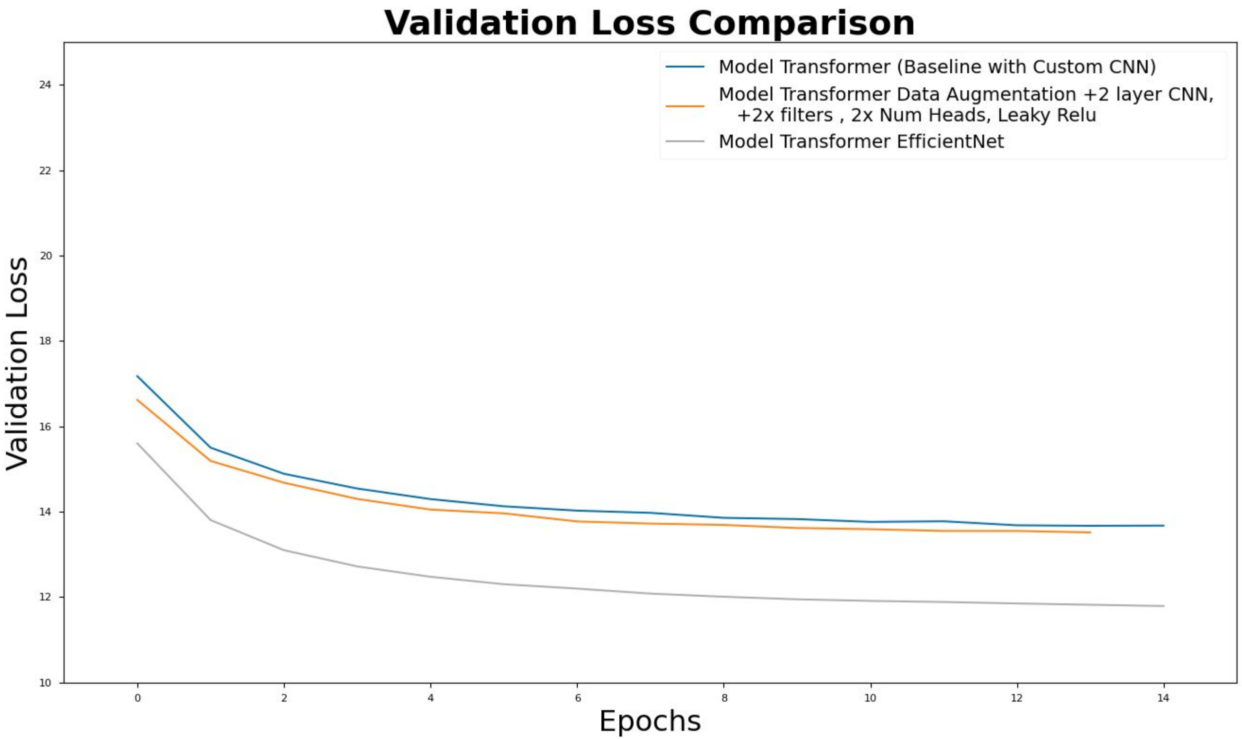
Zoom

Randomly zooms in/out
Factor: ±20% of original
size

Brightness

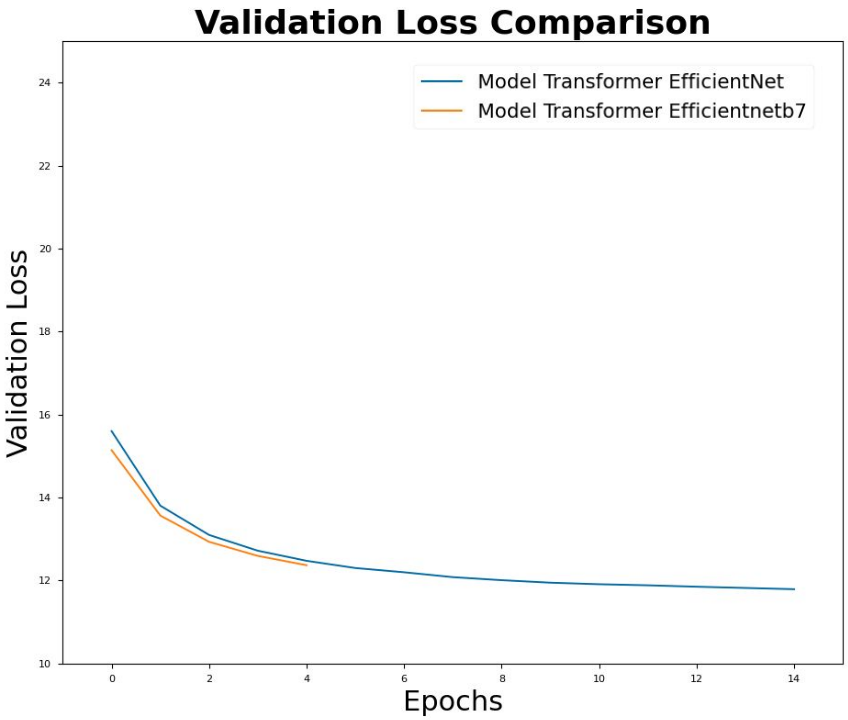
Randomly adjusts
brightness
Factor: ±20% intensity

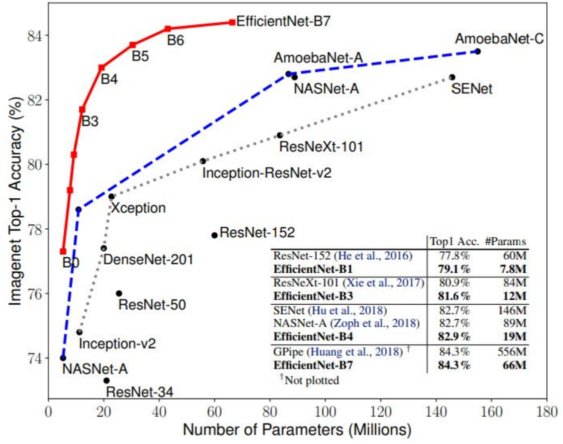
Architecture Optimization



After numerous improvements to the original transformer architecture
Efficient Net remains vastly superior. (4M parameters for both CNN architecture)

Architecture Optimization

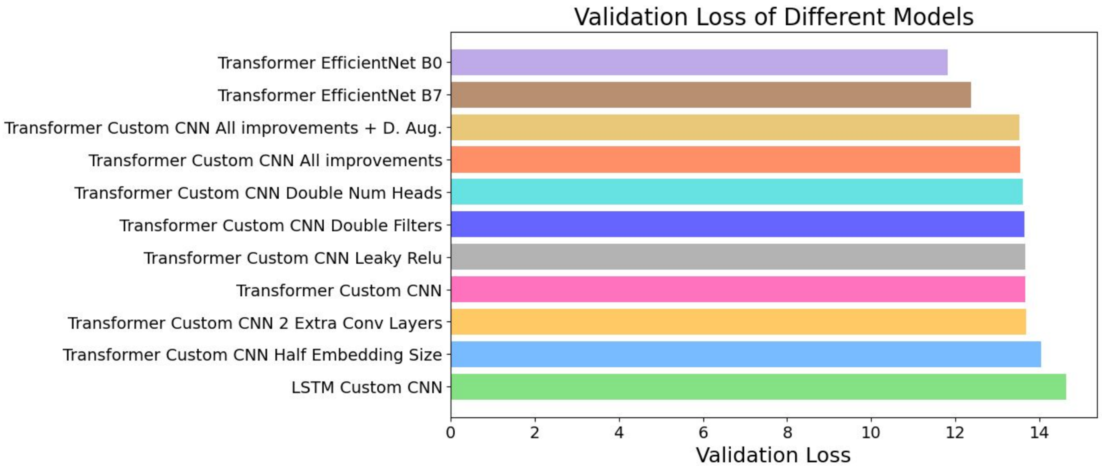


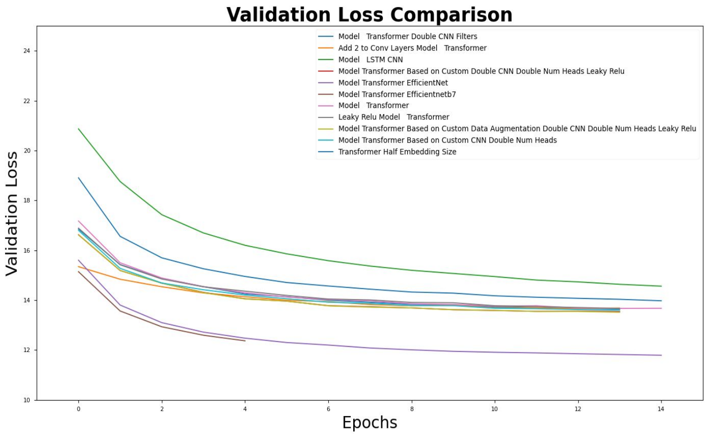


EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks" Mingxing Tan et. al. 2019

EfficientNetB0 due to its small size could be trained for longer and achieve better performance compared to the B7 using 3.2h on **Nvidia T4 GPU**.

Comparison





EfficientNetB0 & EfficientNetB7 **outperform** the other models.

Model Comparison

Model	Epoch	Validation Loss	% Improvement
Model Transformer EfficientNet	14	11.8223	↑ 13.52%
Model Transformer Efficientnetb7	5	12.3706	↑ 9.51%
Model Transformer Based on Custom Data Augmentation Double CNN Double Num Heads Leaky Relu	14	13.5168	↑ 1.13%
Model Transformer Based on Custom Double CNN Double Num Heads Leaky Relu	14	13.5429	↑ 0.94%
Model Transformer Based on Custom CNN Double Num Heads	14	13.6002	↑ 0.52%
Model Transformer Double CNN Filters	14	13.6487	↑ 0.17%
Leaky Relu Model Transformer	14	13.6672	↑ 0.03%
Model Transformer	14	13.6712	0.00%
Add 2 to Conv Layers Model Transformer	14	13.6862	↓ -0.11%
Transformer Half Embedding Size	14	14.0370	↓ -2.68%
Model LSTM CNN	14	14.6381	↓ -7.07%

Metrics

BLEU
N-gram

$$\underbrace{BP(\hat{S};S)}_{\text{Brevity Penalty}} \cdot \exp\left(\sum_{n=1}^{\infty} w_n \ln p_n(\hat{S};S)\right) \qquad p_n = \frac{\sum_{C\in\text{Candidates}} \sum_{n\text{ gram}\in C} \text{Count}_{\text{clip}}(n\text{ gram})}{\sum_{C'\in\text{Candidates}} \sum_{n\text{ gram}'\in C'} \text{Count}(n\text{ gram}')}$$

ROUGE
Longest Common subsequence

$$\frac{\text{Length of LCS}}{\text{Total number of words in the generated text}}$$

PERPLEXITY
Next token Probability

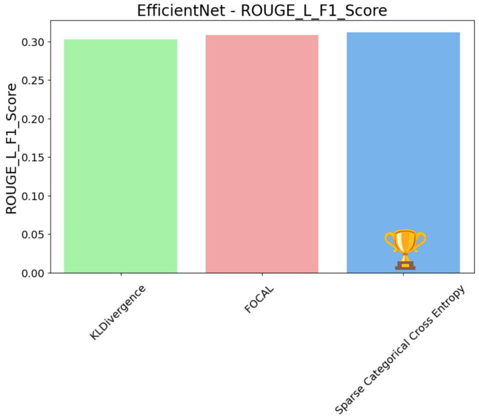
$$e^{-\frac{1}{N}\sum_{i=1}^{N} \log P(w_i|w_1,w_2,\ldots,w_{i-1})}$$

The metrics are Implemented as custom callbacks, as also Model Checkpoint savings and Early Stopping. **Optimizer**: Adam, **Learning rate schedule**: Custom LRSchedule.

Loss Comparison

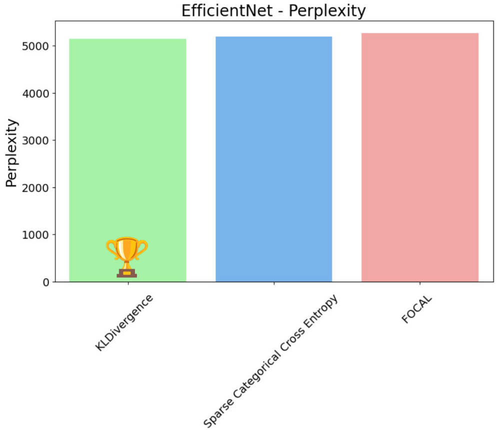
BLEU(4)

Sparse Categorical Cross Entropy 🔝



ROUGE-L (F1)

Sparse Categorical Cross Entropy 🔝



Perplexity

KLDivergence 🔝

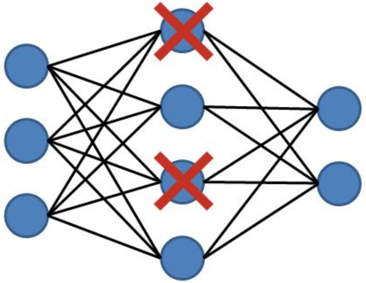
Loss Comparison

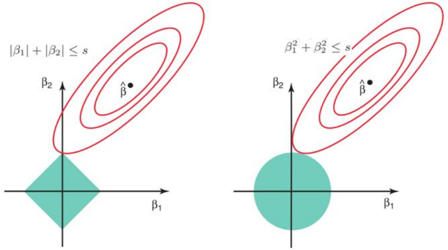
Table 1: Comparison of Different Models and Loss Functions

Model & Loss	BLEU-1	BLEU-4	ROUGE-L Precision	ROUGE-L Recall	ROUGE-L F1	Perplexity	Accuracy
FOCAL - EfficientNet	0.1851	0.0410	0.3595	0.2752	0.3083	5265.7946	0.4703
KLDivergence - EfficientNet	0.1819	0.0385	0.3550	0.2689	0.3028	**5140.9225**	0.4716
Sparse Categorical Cross Entropy - EfficientNet	**0.1894**	**0.0438**	**0.3644**	**0.2783**	**0.3121**	5187.6150	**0.4805**
FOCAL - Custom CNN	0.1348	0.0134	0.2500	**0.1929**	**0.2155**	5373.5072	0.4427
KLDivergence - Custom CNN	0.1316	0.0131	**0.2501**	0.1924	0.2152	5245.2873	0.4468
Sparse Categorical Cross Entropy - Custom CNN	**0.1412**	**0.0170**	0.2413	0.1925	0.2121	**5242.6177**	**0.4479**

Sparse Categorical Cross Entropy, especially with EfficientNet, **outperforms other loss functions** and models across most metrics.

Regularization





Dropout

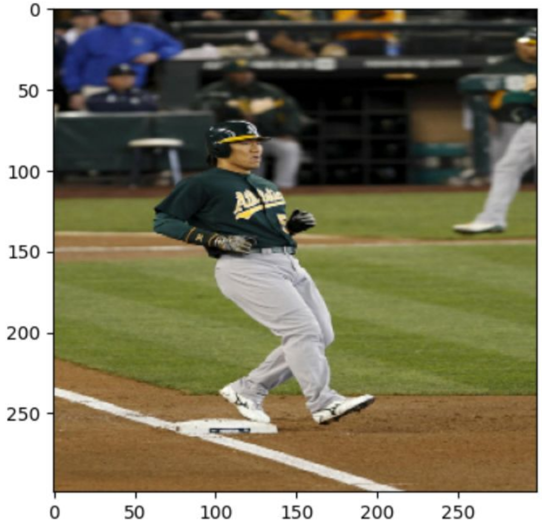
Dropout included in feed forward decoder layers: *Dropout(0.3)*.

Lasso and Ridge Regularization

Not used because there are **no particular signs of overfitting*** to include Ridge or Lasso.

*After trying to reduce the dataset and increase the number of training epochs to overfit the network, adding regularization in all layers of the CNN (L1 with different hyperparameters did not lead to substantial differences).

Prediction Comparison



Model	Prediction	Size
Custom CNN (and improvement)	Two baseball teams playing the baseball game of baseball.	4M CNN + 16M transformer
EfficientNet	Baseball player is getting ready to throw his pitch during baseball game.	4M CNN + 16M transformer
Florence	A baseball player running to first base during a game	200M
PaliGemma	A baseball player, wearing a green and yellow jersey and a black helmet, sprints to first base after hitting a ball...	3000M

Failed & Alternative Experiments

Type	Details
Florence-2 Finetuning	The predicted captions were not semantically correct.
Add more transformer layers	Using more resources, model complexity can be significantly enhanced.
Evaluate other architectures (e.g. GRU)	Explore alternative architectures, such as GRU, to potentially improve performance.

Thanks!

Do you have any questions?

Appendix

Ignore the following slides.

Loss Functions References

[**BLEU**, **Perplexity**] Show and Tell: A Neural Image Caption Generator (Vinyals et al. 2015) *Google*

[**Cross Entropy**] Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering (Anderson et al. 2018) *Microsoft research*

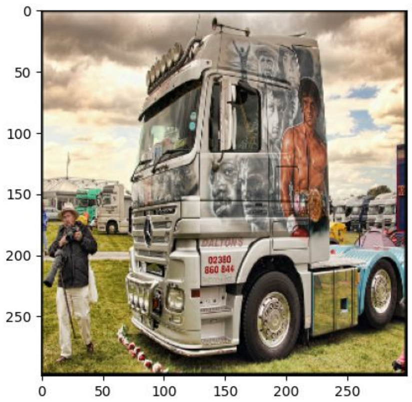
[**BLEU** and **Rouge-L**] Describing like humans: on diversity in image captioning (Wang et al. 2019) *University of Hong Kong*

[**Kullback-Leibler divergence**] Deep Learning Approaches Based on Transformer Architectures for Image Captioning Tasks (Castro et al. 2019) Kumoh National Institute of Technology

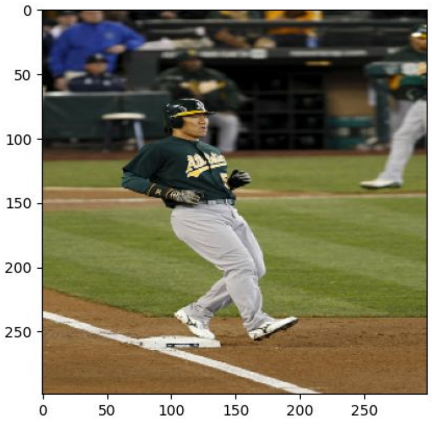
[**Focal cross entropy**] Focal Loss for Dense Object Detection (Lin et al. 2018) *Facebook AI*

Prediction of best Custom Model

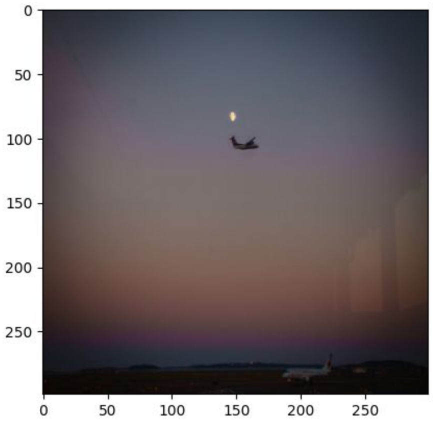
4 M parameter CNN + 16M transformer



Two men standing on the side of the train.



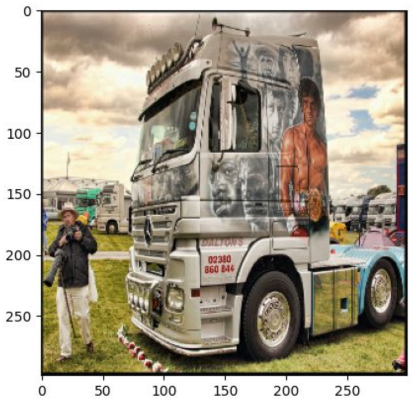
Two baseball teams playing the baseball game of baseball.



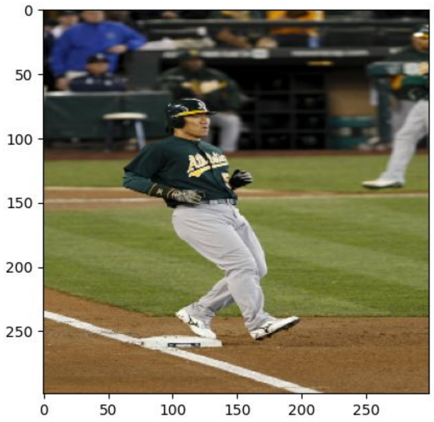
The airplane has landed in the sky.

Prediction of Efficient Net Model

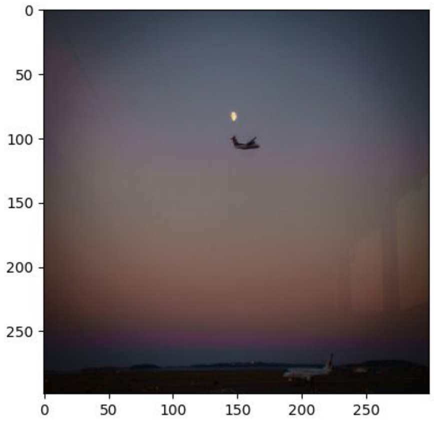
4 M parameter CNN + 16M transformer



The truck has been loaded in the middle of a field.

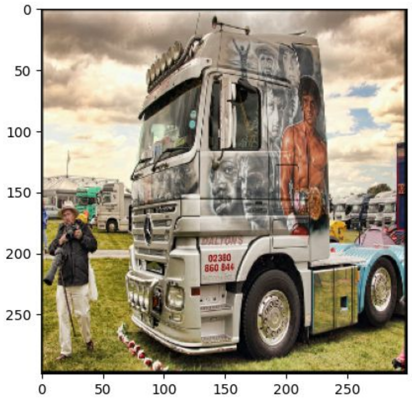


Baseball player is getting ready to throw his pitch during baseball game.

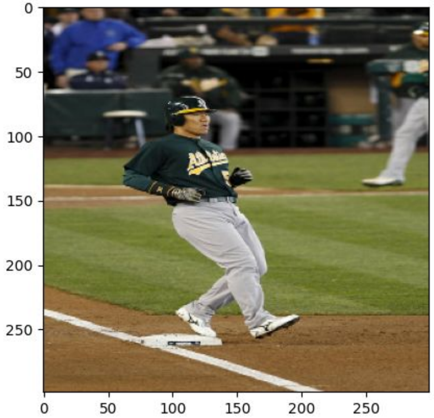


The kite flying over a body with the ocean on a sunny sunset.

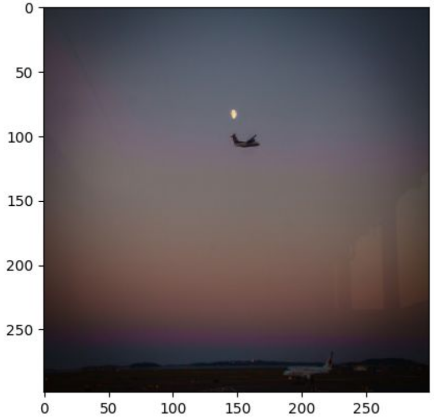
Florence 2 base v0.2b



A large truck with a picture of a man on the side of it.

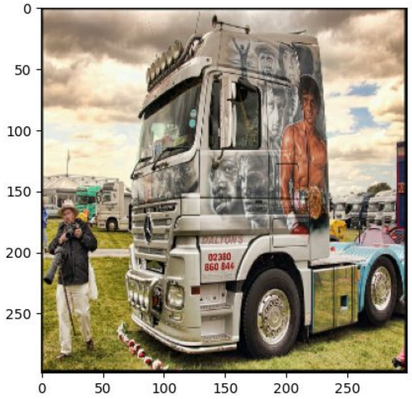


A baseball player running to first base during a game.

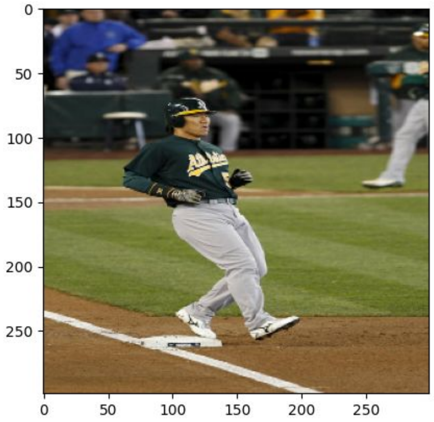


A plane flying in the sky with a full moon in the background.

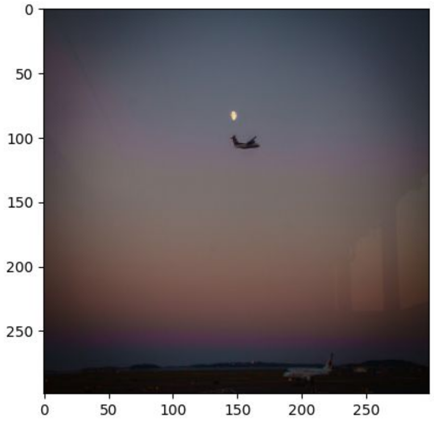
Prediction of PaliGemma 3b



A large truck with a painting of a man on the side, showcasing a variety of details. The truck has a large windshield...

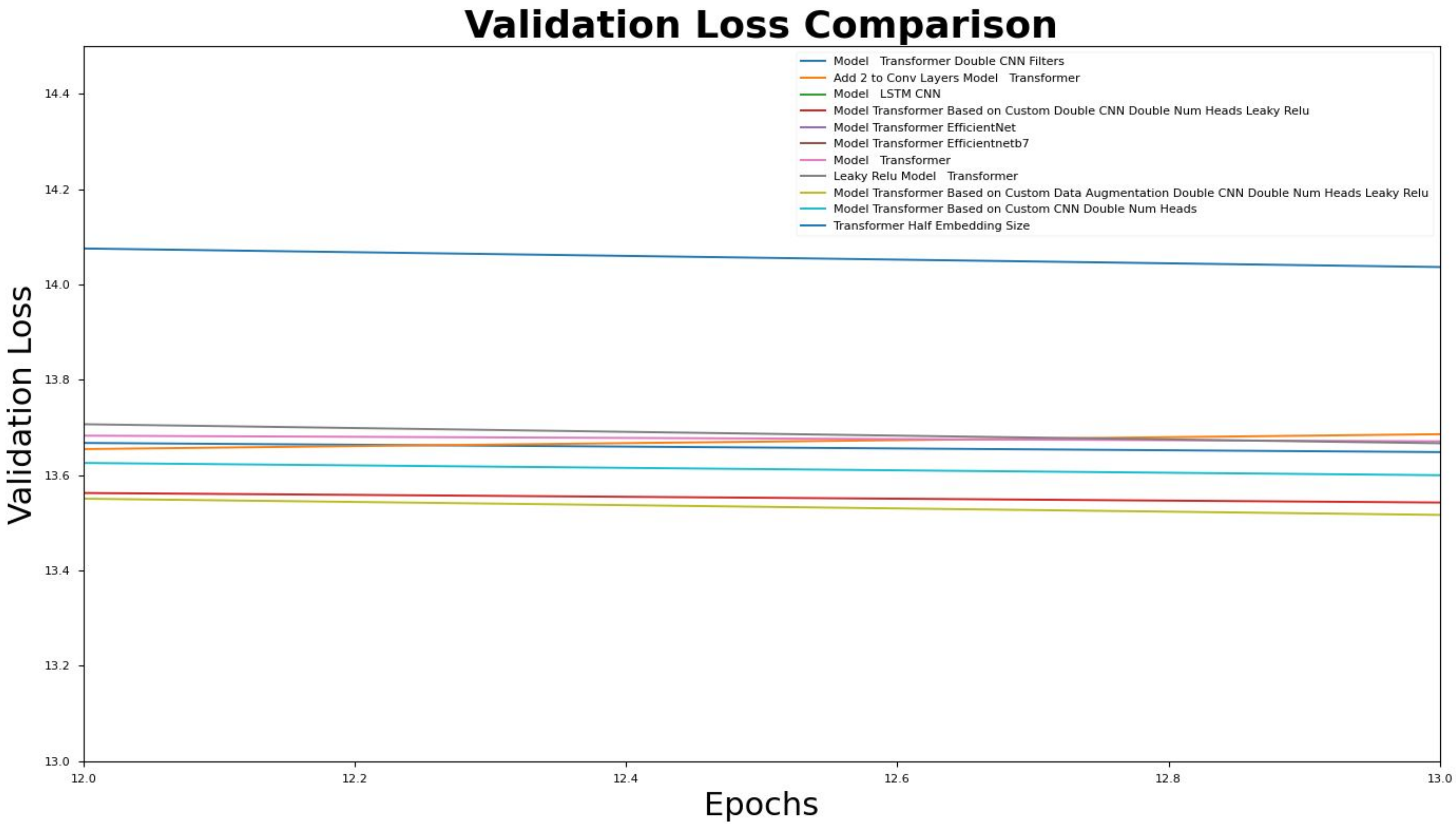


A baseball player, wearing a green and yellow jersey and a black helmet, sprints to first base after hitting a ball...



A plane flies high in the sky at night, its tail shining brightly against the clear sky. The plane is on the ground...

Comparison



Validation Loss Comparison
Model   Transformer Double CNN Filters
Add 2 to Conv Layers Model   Transformer
Model   LSTM CNN
Model Transformer Based on Custom Double CNN Double Num Heads Leaky Relu
Model Transformer EfficientNet
Model Transformer Efficientnetb7
Model   Transformer
Leaky Relu Model   Transformer
Model Transformer Based on Custom Data Augmentation Double CNN Double Num Heads Leaky Relu
Model Transformer Based on Custom CNN Double Num Heads
Transformer Half Embedding Size
Validation Loss
14.4
14.2
14.0
13.8
13.6
13.4
13.2
13.0
12.0
12.2
12.4
12.6
12.8
13.0
Epochs