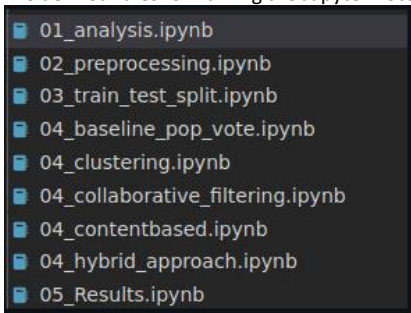


# Thoughts, Decisions and Findings

Freitag, 30. April 2021 14:25

- Project work
  - The team did organized an own stand-up meeting and daily close.
  - Architecture decisions are made by the hole team in the team sessions via slack.
  - Every team member build up it's own Jupyter Notebooks.
  - The project tasks and topics were managed using the page "Project Monitoring" (within that pdf).
  - We used the Cookie-Cutter template for machine learning for our project structure on Github.
- Division of work among the team members
  - Architecture decisions (all)
  - Development of models within Jupyter notebooks (all)
  - Build up presentation and that PDF document (all)
  - Imlementing Attention mechanism (Andras)
  - Preprocessing (Rainer)
  - Metrics Implementation (Thomas)
  - Project management (Thomas)
- Visualizations used within Jupyter Notebooks:
  - Mathplotlib
  - Keras (model plots)
  - Tensorboard (incorporate callback)
  - Store to disk: Models (h5 files) and history of model run (json files)
- Development environment
  - We needed a development environment that supports both large data sets and GPU powered notebooks that may run in background.
  - Therefore, we decided to use Kaggle Notebooks with Kaggle Data sets.
  - We defined rules for naming the Jupyter notebooks (README of Github page [https://github.com/rainerenglisch/aida\\_dfk\\_project/](https://github.com/rainerenglisch/aida_dfk_project/))
  - The team shared the code and development related artefacts via Github repository.
  - Larger files were stored on a google drive (for instance h5 model files).
- Architecture decisions
  - Encoder Part (CNN)
    - As we want to create a sequence of words (caption) from an image, we need a one too many sequence model, that creates a sequence of words from an CNN encoded image vector.
    - Based on our last project "Movie posters", we decided to start with a well pre-trained CNN based on ImageNet data. So the Baseline model uses the InceptionV3 pre-trained CNN, loaded via Keras.
    - Because of low ROUGE-L and BLEU-1 results during testing phase, we decided to add Inception-ResNet-V2 into our models too.
  - Decoder Part (RNN)
    - We decided to compare LSTM and GRU layers in our decoder.
    - For improving the basic models of we decided to implenent the attention mechanism. The Baseline was not equipped with the same.
- Preprocessing
  - Captions:
    - Transformation of sentences to lower case and removal of special characters (comma, semicolon,...).
    - Limitation of vocabulary size to 5000.
    - Image captions are encoded via tokenizer and put into a sequence of tokens. For training, two sequences were created for each caption: One for training input (prepended with a "start" token) and one sequence for the target output (appended with a stop token).
  - Images:
    - In order to save time for encoding the information of a picture, we decided to use transfer learning.(on image net trained CNN). In order to save additional time and ressources image encoding time we saved the encoded images as a numpy files. These numpy files are made available to the Kaggle Notebooks via Kaggle Datasets.
- Findings
  - Transformation of sentences to lower case and removal of special characters (comma, semicolon,...) improved performance of all the models.
  - Limitation of vocabulary size to 5000 improved the performance of all models.

- The models tend to overfit within the first 15 to 20 epochs.  
We suspect that the reason is the relatively small size of the dataset.  
A run with the flickr 30k dataset is recommended by the team.
- The implemented attention mechanism delivered the performance like expected.

# Project Monitoring

Freitag, 23. April 2021 12:49

## Description



## Topics

- Captioning = Ein Bild mit einer Überschrift versehen
  - Anwendung NLP und Computer-Vision
- Datasets
  - 8.000 Pictures
    - manual selected from six different Flickr groups
    - depicting (zeigen) different scenes and situations, NOT well-known people or locations
  - 5 captions (Beschreibungssätze) per Picture, that describe the salient (hervorstechenden) Entities (Objekte, Personen) and Events (Vorgänge)
  - Download location: <https://www.kaggle.com/adityajn105/flickr8k>
    - ◻ Folder holding the pictures
    - ◻ Text file containing the captions
- Mission (Arbeitsauftrag), Deliverables (zu liefernde Artefakte) und Todos
  - ☒ ◻ Preprocessing
    - ☒ ◻ Pictures  
Write features of last hidden layer of InceptionV3 and Inception-ResNet-V2 to Kaggle-Dataset (Rainer)  
Results: Dataset + Jupyter Notebook
  - ☒ ◻ Build an Encoder-Decoder Architecture  
(Encoder: CNN for picture processing, Decoder: RNN for caption processing)
    - ☒ ◻ Baseline Model (Andras, Thomas, Rainer) ohne Attention
    - ☒ ◻ Encoder/Decoder using Attention-Mechanism (Andras)
    - ☒ ◻ Baseline, GRU einsetzen (Thomas)  
~~Encoder Part: Object recognition via multi-class classification, captions transformed to entities for recognition~~
    - ☒ ◻ Encoder/Decoder using Baseline, Variation CNN Part (other Network, other dimension of output) (Rainer, Thomas)
    - ☒ ◻ Encoder/Decoder using Baseline, Variation RNN Part (2 LSTM) (Rainer)
    - ☒ ◻ Encoder/Decoder using Baseline, Variation RNN Part (Glove-Embeddings, ...) (Rainer)
    - ☒ ◻ Encoder/Decoder using Attention Mechanism, Hyperparameter Optimization (learning-rate, layer-units)
      - ☒ ◻ Units of CNN and RNN: 512, 748, 2048
      - ☒ ◻ CNNs: InceptionV3, Inception-ResNet-V2
      - ☒ ◻ RNNs: LSTM, GRU, stacked LSTM
    - ☒ ◻ Build up architecture using Auto-Keras (Rainer), documentation of results backup-page in presentation  
~~CNN/RNN/Reinforcement Learning~~
  - ☒ ◻ Visualize the findings using taught packages
    - ☒ ◻ Matplotlib
    - ☒ ◻ Keras (model plots)
    - ☒ ◻ Tensorboard (incorporate callback)
    - ☒ ◻ Store to disk: Models and history of model run
  - ☒ ◻ Use Metrics (see below)
    - ☒ ◻ Evaluation Function für beide Metriken implementieren (Thomas)  
Output: Library "metrics.py"
  - ☒ ◻ Obtain (erhalte) statistics that corroborate (erhärten, bestätigen) the results
    - ☒ ◻ Loss per epoch
      - ☒ ◻ Training-/Validation-Curves
    - ☒ ◻ Learning rate per epoch
    - ☒ ◻ Metrics only for one run
  - ☒ ◻ Explain the reasons to choosing that statistics  
ROUGE and BLUE Statistics are state of the art for comparison of problems like ours

ROUGE-L takes the sentence structure (sequences of words) into account  
BLUE-1 focused on Unigrams (Words)

- ☒ ☐ Deliverables
  - ☒ ☐ Abgabetermin: 30.04.2021 EOB
  - ☒ ☐ Abgabe aller Artefakte in einer ZIP-Datei mit dem Namen "AIDA2-DKFI-3\_Image-Captioning\_English\_Baligacs\_Schremser.zip"
  - ☒ ☐ Verzeichnis der ZIP-Datei: [https://drive.google.com/drive/folders/1zEQqPRWEcfJ7FFEnEht99b-5rckW4FI\\_Z?usp=sharing](https://drive.google.com/drive/folders/1zEQqPRWEcfJ7FFEnEht99b-5rckW4FI_Z?usp=sharing)
- ☒ ☐ PDF Report in english
  - ☒ ☐ Thoughts an decisions
  - ☒ ☐ Division of work among the team members (Arbeitsaufteilung)
- ☒ ☐ Jupyter Notebooks and related ressources to reproduce the work -> Cookie-Cutter project, without data and Models
- ☒ ☐ Presentation in english 10 - 15 minutes
  - ☒ ☐ Process description
  - ☒ ☐ Approaches used
  - ☒ ☐ Reasons for choosing the approaches
  - ☒ ☐ Results and findings

### Short Sentences

- Reference: "Back to the Future" premiered 30 years ago
- MT: "Back to the Future"

- 1-gram:  $4/4$
- 2-gram:  $3/3$
- 3-gram:  $2/2$
- 4-gram:  $1/1$

$$BP = \begin{cases} 1 & , c > r \\ e^{1-r/c} & , c \leq r \end{cases}$$

$$BLEU = \sqrt[4]{P_1 * P_2 * P_3 * P_4 * BP}$$

- Brevity Penalty:  $e^{1 - \frac{2}{4}} = e^{-1} = 0.37$

- Reference 1: the new movie
- Reference 2: the new film
- MT: the the

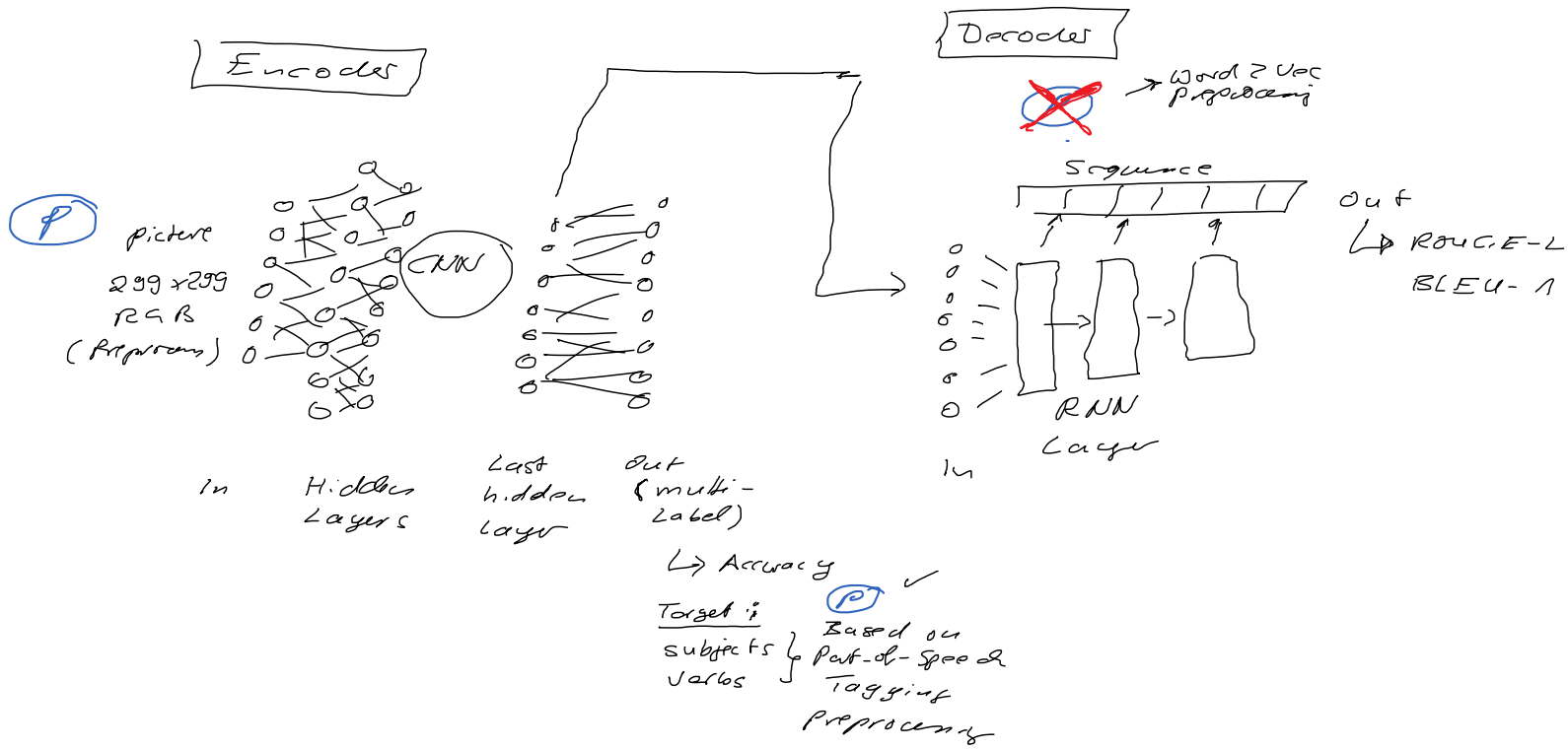
- $P_1 = \frac{2}{4} \cdot \frac{1}{2}$

- Clipping:

- Max count of of n-gram in any reference

# Architecture Draft - High-Level

Montag, 26. April 2021 13:52



**Preprocessing**

# Metrics - Notes and Informations

Dienstag, 27. April 2021 11:11

## Metrics

Overview: <https://stackoverflow.com/questions/38045290/text-summarization-evaluation-bleu-vs-rouge>

- ROUGE-L (Recall-Oriented Understudy of Gisting Evaluation)



<https://ilmoirfan.com/rouge-an-evaluation-metric-for-text-summarization/>

Longest Common Subsequence (LCS) based statistics.

Longest common subsequence problem takes into account sentence level structure similarity naturally and identifies longest co-occurring in sequence n-grams automatically

ROUGE-L – measures longest matching sequence of words using LCS. An advantage of using LCS is that it does **not require consecutive matches but in-sequence matches** that reflect sentence level word order. Since it automatically includes longest in-sequence common n-grams, you don't need a predefined n-gram length.

Vergleich von 2 Sätzen: X = Referenzsatz, Y = maschinell erzeugter und zu vergleichender Satz

ROUGE-L = 0 -> X und Y haben nichts gemeinsam

ROUGE-L = 1 -> X = Y, die beiden verglichenen Sätze sind gleich

Vorteile:

- Es werden nur Sequenzen von Wörtern beurteilt und nicht Wörter einzeln
- Es sind keine vordefinierte Sequenzen von sog. n-gramms notwendig, da der Algorithmus die längsten Wortsequenzen selbst findet

## Precision, Recall and F-measure

To evaluate how accurate our machine generated summaries are we compute the Precision, Recall and F-measure for any of this metric.

In ROUGE **recall** refers that how much words of candidate summary are extracted from reference summary. Formula to calculate recall:

$$R = \frac{\text{Number of overlapping words}}{\text{Total words in reference summary}}$$

For example, recall for unigram in the below example:

R1- The dog bites the man.

S1- The man was bitten by the dog, find in dark.

$$\frac{4}{5} = 0.8$$

It shows that almost all words in candidate (machine generated) summary have been extracted from reference summary. It means our system generated a good summary that is exactly same as reference summary.

But it is not always a good case sometimes the machine generated summary may be too long and contains most of the irrelevant words. So, it may not be a good summary. If the size of machine generated summary is predefined then recall alone may provide the enough information (candidate summary is relevant or not). To find the other case (if machine generated summary is good or not) we also need to compute precision.

In ROUGE **precision** refers that how much candidate summary words are relevant. Formula to calculate recall:

$$P = \frac{\text{Number of overlapping words}}{\text{Total words in candidate summary}}$$

In above example:

$$\frac{4}{10} = 0.4$$

**F measure** provides the complete information that recall and precision provides separately.

$$F - \text{measure} = \frac{(1 + \beta^2) R * P}{R + \beta^2 * P}$$

$\beta = 1$  so.

$$F_1 = 2.0 * \frac{0.8 * 0.4}{0.8 + 0.4} = 0.53333$$

Beispiel:

X = police killed the gunman

Y1 = police kill the gunman -> ROUGE-L (bei  $\beta=1$ ) =  $3/4 = 0,75$  -> longest sequence: police the gunman

Y2 = the gunman kill police -> ROUGE-L (bei  $\beta=1$ ) =  $2/4 = 0,5$  -> longest sequence: the gunman

In Y1, the first word and last two words match the reference, so it scores  $3/4$ , whereas

Y2 only matches the bi-gram, so scores  $2/4$ .

Aus <<https://stats.stackexchange.com/questions/301626/interpreting-rouge-scores>>

Formula for Presentation:

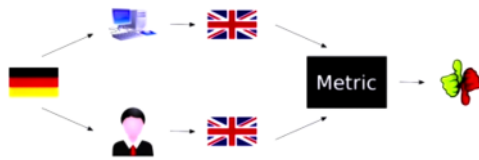
$$\text{ROUGE - L Recall} = \frac{\text{Number of overlapping words between reference and candidate sentence}}{\text{Total number of words in candidate sentence}}$$

- BLEU-1 (bilingual evaluation understudy)

BLEU-1 sagt aus, dass der Score für Unigramms (1-gram) betrachtet wird?

Es wird die maschinelle Übersetzung mit menschlichen Übersetzungen verglichen.

Hierbei werden allgemeingültige Satzsegmente im maschinelle Text mit qualitative guten Referenzsätzen (mehrere!) der menschlichen Übersetzung verglichen.



BLEU = 1, sehr gute Übereinstimmung

BLEU = 0, keine Übereinstimmung

Beispiele:

<https://www.coursera.org/lecture/machinetranslation/bleu-Bv81F>

- Reference: "Back to the Future" premiered 30 years ago
- MT: "Back to the Future" had premiered 30 years ago

1-gram: 8/9

2-gram: 6/8

3-gram: 4/7

4-gram: 2/6

Geometric Mean:  $\sqrt[4]{8/9 \cdot 6/8 \cdot 4/7 \cdot 2/6}$

*Handwritten notes: "nur MT" next to 2-gram, "betrachtet" next to 3-gram.*

Formula for presentation:

$$BLEU - 1 \text{ Precision} = \frac{\text{Number of overlapping words between reference and candidate sentence}}{\text{Total number of words in candidate sentence}}$$

Reference 1: students said they looked forward to his class

MT: students said they were excited about his lecture

$P_4 = 0/8$

Document level scores:

Aggregate statistics over whole document

## BLEU

- Matches exact words
  - Several references possible
- Adequacy: Modeled by word precision
- Fluency: Modeled by n-gram precisions
- No recall: *only precision is calculated*
  - „brevity penalty“ to prevent short sentences
- Calculate aggregate score over a large test set

**n-gram** is a contiguous sequence of  $n$  items from a given [sample](#) of text or speech. The items can be [phonemes](#), [syllables](#), [letters](#), [words](#) or [base pairs](#) according to the application

Aus <https://en.wikipedia.org/wiki/N-gram>

Implementierung ROUGE-L:

Python:

<https://pypi.org/project/rouge-score/>

<https://pypi.org/project/easy-rouge/>

<https://pypi.org/project/rouge-metric/>

<https://pypi.org/project/py-rouge/>

<https://pypi.org/project/rouge/>

Implementierung BLEU:

Python

<https://pypi.org/project/bleu/>

NLTK:

<https://machinelearningmastery.com/calculate-bleu-score-for-text-python/>

<https://stackoverflow.com/questions/32395880/calculate-bleu-score-in-python>

Keras:



Es muss eine sog. Custom-Metric angelegt werden:

<https://keras.io/api/metrics/>

Implementierung der von uns benötigten Metriken für Keras:

<https://github.com/danieljl/keras-image-captioning>

Anwendung:

```
import src.models.metrics as met
```

```
...
```

```
model.compile(optimizer=Adam(lr=self._learning_rate, clipnorm=5.0),  
              loss=categorical_crossentropy_from_logits,  
              metrics=[met.categorical_accuracy_with_variable_timestep])
```

```
...
```

## Summay Results

Freitag, 30. April 2021 19:18

Model	Encoder (CNN)	Decoder (RNN)	Run	ROUGE-L recall TEST	BLEU-1 precision TEST	ROUGE-L recall TRAIN	BLEU-1 precision TRAIN
Baseline	InceptionV3 (512 units)	LSTM (512 units)	20210429-080430	0,3578	0,4008	0,5149	0,5901
InceptV3_GRU_NoAttention_CustEmbedding	InceptionV3 (512 units)	GRU (512 units)	20210429-131938	0,3646	0,4036	0,5093	0,5794
InceptV3_LSTM_Attention_CustEmbedding	InceptionV3	LSTM		0,4537	0,4464	0,4520	0,4401
InceptV3_GRU_NoAttention_CustEmbedding_768-units-all	InceptionV3 (768 units)	GRU (768 units)	20210429-173000	0,3639	0,4028	0,5415	0,6102
Baseline + Glove Word vectors	InceptionV3 (2048 units)	LSTM (2048 units)		0,3592	0,3968	0,5893	0,6715
Baseline + Glove Word vectors + 2x LSTM	InceptionV3 (2048 units)	LSTM (2048 units)		0,3515	0,2503	0,4028	0,2631
Inception-ResNet-V2_GRU_NoAttention_CustEmbedding	Inception-ResNet-V2 (512 units)	GRU (512 units)	20210429-224955	0,3704	0,3944	0,3990	0,4339
Inception-ResNet-V2_GRU_NoAttention_CustEmbedding	Inception-ResNet-V2 (512 units)	GRU (512 units)	20210430-201023	0,3624	0,4401	0,4934	0,4934