Data and Preparation

General:

- data we need to collect has to be both unimodal and multimodal
  - multimodal obvious -> needed to align modalities, as described in e.g. section about "(See, Hear, and Read:) Deep Aligned Representations"
  - multimodal means in this case dataset of image-text pairs
  - unimodal -> needed for first tests and poc of distillation process
  - unimodal data also needed for evaluation and comparison of unimodal models from research papers, like Data2Vec, as well as comparison between unimodal and multimodal distilled models (models of this thesis)
  - unimodal data also needed for stage-wise knowledge distillation in section about "Mixing Positional Encodings" (will be elaborated on in the respective section)

Data Selection and Collection:

- starting with unimodal:
- collecting unimodal data is not a problem, many highly curated and large datasets available
- because we are using Knowledge-Distillation based on self-supervised distillation, so we do not need labels for the distillation process, we can use any image dataset
- for image data, we select imagenet
  - build for image classification and object detection (labled, but, again, not necessary)
  - each image corresponds to one of 1k classes
  - very popular, high quality, high variety -> 1000 classes, by standards of SOTA models it is a medium sized dataset (ca. 1.2M train images)
    - why medium sized? -> papers used in this thesis have been trained on much larger data -> VLMo around 14 million image(-text) examples,

    BEiT on more than 35M, and FLAVA (only mentioned a couple of times) even 70M
    - models also much larger than models build here, so we do not need as much, nor is it feasible for us to train on that much data
  - Data2Vec Image model exclusively trained on imagenet
  - Data2Vec, BEiT, VLMo, and FLAVA all use imagenet for evaluation -> we should use it as well
  - we use the full dataset of the 2012 version, with 1.2M images for training and 50k for validation
  - during pretraining, i.e. the Knowledge-Distillation, we apply the same data augmentation as in Data2Vec2
    - as we also use Data2Vec2 as the teacher in many experiments, this allows us a close comparison between the models, as they are trained on the same data
    - data augmentation includes random resized crop, followed by a random horizontal flip, followed by normalization each channel seperatly using channel wise mean and standard deviation computed from the imagenet training set -> standard procedure for image preprocessing, and used for all images throughout this work
    - random resized crop: crop a random part of the image, then resize it to the desired size (224x224)
      - crop size is hyperparameter, but we just use the same as in Data2Vec2, which is 0.08 to 1.0 of the original image size

    - 8% as lower bound seems to be a very low value, a lot of information is lost, but it is a common value in the literature, so we do

    the same
  - random horizontal flip: randomly flip the image horizontally (self-explanatory)
- for validation, we resize each image to the same size as in training (224x224) and normalize it using the same procedure as in training
- generally ALL images in this thesis will be scaled to the same size: 224x224
- we access the data from Huggingface's dataset hub

- benchmarks published on downstream tasks/dataset like GLUE, more on that later
- data is not that much and only meant for fine-tuning and benchmarking as downstream task
- as with images, we a just trying the replicate the outputs, i.e. the representations of the input data
- we can use any text (dataset(s)), as long as it is large enough
- for text data (pretraining, which is the Knowledge-Distillation that we do) we select openwebtext
  - dataset build to reproduce datasets used to train GPT-2
  - publicly available and popular, used by e.g. BEiT3
- we access the data from Huggingface's dataset hub
  - published as slices and without any split
  - we take subsets 0-4 for training and 5 for validation, which is about 25% of the data
- due to open source efforts, data is already preprocessed and cleaned
- we apply further preprocessing by removing empty lines and null bytes, which we found are quite common and lead to problems during encoding and training, as they provide no learnable information
- the text of every dataset, containing text, so also openwebtext, is tokenized and encoded using the GPT-2 byte-pair encoder (citation here -> same as in D2V), with a vocabulary size of 50262 tokens
- we seperate the sentences using the end-of-sentence token, also done by Data2Vec2
- also used by Data2Vec2, and we use it, again, for the purpose of comparison
- so save disk space, we save the training and validation sets of owt in a single binary file, respectively
  - the binary files already contain the encoded text, so that we only need to batch them during training
  - in order to ensure correct encoding and to save the time for implementing the binary encoding, we use the dataset functionality of Fairseq, a library for sequence-to-sequence models developed by Facebook/Meta, to encode an binarize the text data, which is also used by Data2Vec2
- in total, our openwebtext subset consists of more than 2.5 billion tokens and consumes roughly 6 GB of disk space
- really low, compared to the image data, which is about 150 GB
- we do not use bookcorpus and english wikipedia, datasets Data2Vec2 was trained on, as Openwebtext appears to be a more recent and popular dataset for Knowledge-Distillation -> DistilGPT2 and Distilroberta trained on openwebtext, results showed that this dataset yields good results for (knowledge) distillation

- multimodal data:
- we need to use datasets with image-text pairs
- as Data2Vec not multimodal, we do not have any reference datasets
- however, many multimodal models, like BEiT and VLMo use the same popular multimodal dataset
- we therefore also opt for them
- we use COCO, Visual Genome, and a subset of Google's Conceptual Captions
- even though COCO contains just contains 82783 images, which is not that much, it contains multiple captions per image, meaning we can create multiple image-text pairs from one image
- images have a little more than average 5 captions, yielding a total of 566747 actual examples for the training set (used for Knowledge-Distillation)
- same goes for Visual Genome: contains 108249 images
- here, each image has on average even 50 captions yielding 5408689 unique image-text pairs (examples) for training
- we have to consider why that is -> 50 captions quite a lot per image
  - the reason why that is is because captions are extracted from region descriptions of images
  - describe, not as in coco, the focus of the image, or rather a short description/summary of the image, but individual, sometimes

  small, regions of the image -> VG also used for object detection
- three problems can come with that:
  1. the captions can be very similar, and we sometimes observe repreated captions
  2. captions might focus on small parts of the image, failing the describe the actual focus/context of the image
     - not a problem for object detection, but for our purpose it is not ideal
     - aligning representations of image with very short and specific caption about a small portion of the image might confuse the model -> focuses less on the general context/content
  3. the ratio of unique text and images heavily skewed towards text, image part of our multimodal models might lack behind their text counterparts, leading to worse performance when aligning the modalities
- solution:
  - discard repeated captions
  - concatenate captions of the same image until we reach the maxiumum sequence length of the model we are training
  - if a caption does not fit anymore, we start a new caption
  - will reduce the number of captions we have per image, but captions will be longer
    - model has more information available -> description is more detailed
    - might "caption" the content of the image better
    - we reduce the dominance of unique text over unique images
    - since single region descriptions are often very short, we would pad a lot of timesteps to reach the maximum sequence length
      - padding is a source of inefficiency, as it is not learnable information, but just a placeholder
    - through concatenating them we make optimal use of the maximum sequence length
  - how do we concatenate them? -> we simply concatenate them and separate them by the "and" token to create coherent sentences, even though they might not be gramarically correct

- disadvantage: reduces the number of unique examples the model sees during training
- that is why we do not do the same for COCO, as the ratio is more balanced
- we also additionally use a subset of Google's Conceptual Captions, which originally contains over 3.3M unique image-text pairs
- here each image is only associated with one caption, reducing the relative overrepresentation of text
- Google only provides an index file with captions to image urls
  - urls are sourced from the web, so there is no guarantee that all images are still available
- we use a subset of 400,000 randomly selected and, as of June 2024, available images (with their captions)
- storing the whole dataset is not feasible for us and the combination with COCO and VG should already provide enough data for training
- ids and urls of image-text pairs used are available on GitHub

Dataset	# Images	Avg. Captions	Avg. Caption Length	# Image-Text Pairs
COCO	82,783	5.0	11.0	566,747
VG	108,249	50.0	4.7	5,408,689
VG Concat	108,249	6.0	52.7	653,792
CC3M Subset	400,000	1.0	50.0	400,000

Table 1: Multimodal Dataset used for aligning Image and Text. The maximum text sequence length is, inspired by BEiT3, set to 64 tokens. The concat version of VG will therefore also have a maximum text sequence length of 64 tokens.

- all captions are tokenized and encoded using the same GPT-2 byte-pair encoder as the text-only data
- as usual, and done in BEiT, VLMo, and FLAVA, we prepend each caption with a start-of-sequence token and append an end-of-sequence token
- we use the same data augmentation as in the unimodal case for the images
  - that is, during training we apply random resized crop, followed by a random horizontal flip, followed by the imagenet normalization
  - during validation, we resize the image to 224x224 and normalize it using the same procedure as in training
  - the only difference lies in the crop size of the images
  - min crop size set to 0.08 for image pretraining (unimodal image distillation)
  - means at a minimum we could crop 8% of the image, and discard the rest
  - destroys a lot of information, not a problem for image only training, but when the image has a caption describing the image, and some parts focusing on the cropped parts, which can especially happen for VG, where the captions consists of region descriptions, then we might have captions that do not match the image anymore
  - that is why papers that use random crop use higher values -> BEiT3 uses 0.5, FLAVA 0.9, VLMo uses RandAugment
  - we consider 0.9 too high and 0.5 too low, so we opt for 0.6