TOWARDS MAKING TRANSFORMER-BASED LANGUAGE MODELS LEARN HOW CHILDREN LEARN

by

Yousra Mahdy

A thesis

submitted in partial fulfillment

of the requirements for the degree of

Master of Science in Computer Science

Boise State University

August 2022

BOISE STATE UNIVERSITY GRADUATE COLLEGE

DEFENSE COMMITTEE AND FINAL READING APPROVALS

of the thesis submitted by

Yousra Mahdy

Thesis Title: Towards Making Transformer-based Language Models Learn How Children Learn

Date of Final Oral Examination: 12 April 2022

The following individuals read and discussed the thesis submitted by student Yousra Mahdy, and they evaluated the student's presentation and response to questions during the final oral examination. They found that the student passed the final oral examination.

Cassey Kennington, Ph.D.	Chair, Supervisory Committee
Francesca Spezzano, Ph.D.	Member, Supervisory Committee
Tim Anderson, Ph.D.	Member, Supervisory Committee

The final reading approval of the thesis was granted by Cassey Kennington, Ph.D., Chair of the Supervisory Committee. The thesis was approved by the Graduate College.

ABSTRACT

Transformer-based Language Models (LMs), learn contextual meanings for words using a huge amount of unlabeled text data. These models show outstanding performance on various Natural Language Processing (NLP) tasks. However, what the LMs learn is far from what the meaning is for humans, partly due to the fact that humans can differentiate between concrete and abstract words, but language models make no distinction. Concrete words are words that have a physical representation in the world such as "chair", while abstract words are ideas such as "democracy". The process of learning word meanings starts from early childhood when children acquire their first language. Children learn their first language through interacting with the physical world by using simple referring expressions. They do not need many examples to learn from, and they learn concrete words first from interacting with their physical world and abstract words later, yet language models are not capable of referring to objects or learning concrete aspects of words.

In this thesis, I derived motivation from the way children acquire language and combined a concrete representation of certain words into LMs while leveraging its existing training regime. My methodology involves using referring expressions to visual objects as a way of linking the visual world representations (images) with text. This takes place by extracting word-level visual embeddings for concrete words from images, while extracting word-level contextual embeddings for abstract words from

text and then using them to train language models. In order to enable the model to differentiate between concrete and abstract words, I use a dataset that gives an indication of the level of concreteness for words to determine how information about each word was applied during training.

The work presented in this thesis is evaluated using a standard language understanding benchmark by analyzing the effect of using the proposed training regime on the language model and comparing its performance with traditional language models trained on large corpus data. In the final analysis, the results demonstrate that using referring expressions as the input text to train language models yields better performance on some language understanding tasks than using traditional, corpus-based text. However, the proposed approach cannot affirm that adding visual knowledge and/or concreteness distinction knowledge enriches LMs.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1:

INTRODUCTION

Transformer-based language models (LMs), like the BERT model (Devlin *et al.*, 2019), have drastically changed the field of NLP. These models are very powerful in that they follow a pre-training/fine-tuning approach. They pre-train on large-sized unlabeled text data and can then be fine-tuned on smaller task-specific text data, using the pre-trained knowledge of word contextual meanings. For example, the BERT language model is pre-trained on the whole English Wikipedia and the Brown corpus, then it can be applied to at least a dozen NLP tasks such as sentiment analysis, natural language inference, text classification and paraphrasing tasks. Results show that BERT works significantly better than prior models, even the task-specific models. The BERT model is pre-trained on large amounts of text, using the regime given sentences with randomly masked (i.e., hidden) words, guessing the words that were masked, and given an input sentence guessing the sentence that follows. After the pre-training step, the model learns a degree of syntactic and semantic relations of words encoded as embeddings, which is further used for downstream tasks.

However, being trained only on textual data, these models assume that all the words are *abstract*, which is different from how humans learn and understand natural languages. Humans begin by learning *concrete* words, which are words that have physical representation in the world such as *dark*, *light*, *red* and *chair*; later on, they

learn abstract words which are ideas that cannot be observed in the physical world such as *freedom*, *grace* and *democracy*. Since LMs learn only from text, they are unaware of the concrete meaning of words and incapable of connecting a word that should have a concrete meaning to the objects in the world that it derives its meaning.

The question remains, "Are these LMs useful in all cases where language is used?". Rogers *et al.* (2021) make the observation that, despite their successes, there is a lot of hype surrounding them, including claims that language models learn to encode deep semantic meaning and "understand" language. Furthermore, Liang *et al.* (2021) make another observation that the meaning of a word is different when it is learned from images or text. This implies that in order to have language models that form a holistic understanding of human languages, they need to learn from both the text and physical world representations (i.e., images, sounds, videos, emotions). Moreover, I think it is important to take inspiration from how children learn a language, which is nowhere near how LMs are trained. According to child development literature, children start learning the language before being exposed to text. They are able to learn with small amounts of experience with the physical world as the child's development is multi-modal in nature (Smith & Gasser, 2005). Additionally, they learn concrete words first from interacting with their physical world, and then they learn more abstract words as they get older (Borghi *et al.*, 2019). Also, children do not need massive text data to learn a language as they use *referring expressions*, which are sentences used to refer to objects in the physical world, such as: "The blue ball" and "The Lady with the white dress". Notably, the task of learning a language using referring expressions, is considered a critical developmental phase in language learning for children (McCune, 2008).

The aim of this work is to explore an area where LMs can be improved, using the motivation derived from the way children acquire languages. The proposed approach adds a concrete representation of certain words to the ELECTRA LM (Clark *et al.*, 2020), while leveraging its existing training regime; hence, the model acquires knowledge of the physical world, which narrows the gap in understanding word meanings between LMs and humans. Moreover, this work opens the door for using a smaller number of text examples to train LMs, which can be utilized within applications that have a minimal amount of data and the interaction with the physical world is needed. For example, the robotics applications where large LMs (like BERT) cannot be used due to their limitations (i.e., trained on text-only using a huge training dataset).

In the next section, I explain the research question and hypothesis that bridges these gaps. For the following chapter, "Background and Related Work", I provide additional detailed background on language models, specifically BERT, ELECTRA and WAC models. Additionally, I illustrate the data used to approach the research question, and the metrics used to evaluate the proposed approach. In the "Related Work" section, the contributions that tackled the same or close area of interest are explored. Furthermore, in the "Method" chapter I explain the model architecture, the data pre-processing steps, the training regime, and the experiments performed throughout this thesis to answer the research question. Lastly, in the "Evaluation" chapter, the results, analysis, limitations and the final contribution of this work are presented and discussed.

1.1 Thesis Statement

Can we move towards emulating the settings in which children learn their language to enrich language models? On a more technical level, if language models are given knowledge of the physical world, with a distinction between concrete and abstract words and a dataset that is similar to a child development-inspired dataset; will it help with better model performance with less amount of pre-training data?

I hypothesize that using a training approach that closely mimics children's learning methodology (i.e., I give the language model access to a representation of the visual, concrete world) and using a child-development inspired dataset (i.e., resolving referring expressions to visual objects)— with a distinction between concrete and abstract words, will arrive —to some degree— at better results sooner than using the traditional training approach with comparable data size (measured in Bytes). By sooner I mean that a smaller amount of text data would give us better performance when compared to a traditional language model, pre-trained using abstract knowledge (i.e., text) alone.

These questions are approached by performing a set of experiments and analyzing its results in depth using the methodology and tools illustrated in the next chapters.

CHAPTER 2:

BACKGROUND AND RELATED WORK

Overall, many contributions have been made to improve the natural language understanding task using language models. Starting from representing words in a form that can be processed and manipulated by different models, which is known as word embeddings. Other contributions are more concerned with capturing the distributional meaning of words; preliminary work concentrated on extracting textual semantics to understand the natural language. This task has been drastically improved by the transformers architecture through self-attention (Vaswani *et al.*, 2017). Such models as ELMO (Peters *et al.*, 2018) and GPT (Radford & Narasimhan, 2018), are both based on the transformer architecture and are both pre-trained on massive amounts of text data. These models can be further extended to be used on more specific tasks, with some modifications. The ELMO model captures the context of a word from both directions (left to right and right to left) via independent networks and concatenates them to get a loose overall representation of the context, and after pre-training, the model can be applied to further NLU tasks by using pre-defined different architectures than what it is pre-trained on. Alternatively, the GPT model is unidirectional (left to right). But unlike the ELMO model, the GPT model can be fine-tuned on NLU tasks with minor modifications to the architecture.

In the following sub-sections, I explain more about the cornerstones that my work

is based on; BERT model, ELECTRA model, WAC model, RefCOCO dataset,VisDial dataset and the newly developed ELECTRA + WAC model. Lastly, I explain the metric I use to evaluate my work, which is the GLUE Benchmark.

2.1 BERT

The Bidirectional Encoder Representations from Transformers (BERT) model (Devlin *et al.*, 2019) is a revolutionary model that elevated performance while using the transfer learning approach in language models. This was achieved through pre-training on a large text dataset, then fine-tuning on various downstream tasks. Remarkably, the BERT model outperformed task-specific models and other transfer learning models like GPT. One main point of strength of the BERT model is using a bidirectional pre-training approach instead of left-to-right or right-to-left attention approaches. As a result, it captures word representations (i.e., the meaning of words) from both left-to-right and right-to-left contexts.

2.1.1 Input Representation

The input representation for the BERT model is designed to support the model with pre-training and fine-tuning tasks. The input embeddings consist of three layers summed together, as shown in Figure 2.1. The first layer is the token embeddings layer. A token is the smallest building block for a sequence. As a sentence consists of consequent textual words, a sequence is consequent tokens; a word in a sentence is converted to one or more tokens. For example, the word “painting” can be tokenized into “paint” and “##ing” tokens, the “##” special token that precedes the “ing” part of the word “painting” denotes a token that is not extracted from the beginning of a word. All tokens are added to a list of tokens, which is usually referred to by the BERT Vocabs. There are special tokens used by the BERT model. Such tokens

as: the [CLS] token; placed at the beginning of sentences, the [SEP] token; used to separate two sentences or at the end of the sentence —based on the pre-training or fine-tuning task—, the [MASK] token; which masks an input token, and the [UNk] token; which is given to any unknown word that might appear in the training dataset, although it is not recommended to deal with unknown tokens by using the [UNK] token; instead, unknown tokens should be added to the BERT Vocabs list of tokens.

The second embedding layer is the segment embeddings layer. This layer differentiates between sentences in the same sequence, as tokens within the same sentence have the same segment embeddings vector, while tokens from different sentences have different values for the segment embeddings vector.

The third embeddings layer is the position embeddings layer, which carries the information about the position of a token within the input sequence. The information encoded within this layer is crucial to the BERT model's pre-training, as the self-attention architecture in BERT is indifferent of the position of tokens within the context (Vaswani *et al.*, 2017). Consequently, if such information is not added as an input to the model, it cannot retrieve it during the pre-training process. Furthermore, the model cannot learn the structure of the language. The total input representation to the BERT model is the sum of all three embeddings layers.

The WordPiece Tokenizer

The WordPiece tokenizer (Wu *et al.*, 2016) is the tokenizer used to create the BERT Vocabs list of tokens. It is a sub-word tokenization algorithm that is used to divide words into sub-words (tokens). This tokenizer takes a textual dataset as input and returns a list of common tokens as output. The WordPiece tokenizer starts at the letter level as a token, then combines tokens together if they appear more

than a pre-defined threshold number of times. Hence, it creates the most common sequences of tokens as separate tokens. A token at the start of the word is used without any special characters ahead of it. However, a token that is not the first part of the word is preceded by the "##" string. A word can be converted to a single or several tokens. The BERT Vocabs list has 30,522 tokens, with the first 1000 tokens as [unused] tokens, which can further be replaced with user-defined tokens if a user wants to add more tokens that are not in the original vocab file.

2.1.2 Pre-training Data

The BERT model is pre-trained on corpus text data extracted from two sources: The book corpus dataset (Zhu *et al.*, 2015), and the English Wikipedia (the long contiguous sentences only), with a total of 3.3M words. These datasets are extremely large, and they are extracted from different sources which enriches the coverage of this dataset, and adds a more broad perspective for a large number of topics.

2.1.3 Pre-training Tasks

The BERT model pre-train using two tasks: The Masked Language Model (MLM) task and the Next Sentence Prediction (NSP) task. Both work towards enriching different parts of the model which is illustrated next.

Masked Language Model(MLM)

This task is responsible for the bidirectional training in the BERT model. The way it works is that a certain percentage (15%) of the input sequence tokens are masked, and the model is pre-trained to predict the true identity of the masked tokens. The MLM task has the following settings: for 15% of the input sequence of tokens, only 80% of this percentage is replaced with the [MASK] token, 10% of the

time it is replaced with a random token and 10% is kept the same. This is to adapt the model to work with different fine-tuning tasks, as the [MASK] token is not used in fine-tuning tasks (predicting the identity of masked tokens is not a common NLU task). According to the ablation studies presented in the BERT paper (Devlin *et al.*, 2019), this task is the key source of the high-performance BERT accomplished over prior literature.

Next Sentence Prediction(NSP)

The NSP task aims to improve performance for sentence-based fine-tuning tasks such as Question Answering (QA) and sentence entailment, etc. In this task, for a sentence in input, another sentence is drawn randomly from the whole dataset and the model task is to recognize if the next sentence is either [isNEXT] or [notNext]. According to the ablation studies done by Devlin *et al.* (2019), this task improves the model performance from 87.9 to 88.5 (BERT-base) on the GLUE benchmark.

2.1.4 BERT Drawbacks

The BERT model proved significant improvement using the GLUE benchmark, such as the similarity between sentences, grammatically correct sentences, sentiment (positive, negative, or neutral), etc. However, the model is data hungry and it is a large-sized model; the BERT-base is 110M parameters and BERT-large is 340M parameters. Additionally, the model is constrained by 15% of tokens to calculate the loss; hence, it needs to be trained on a huge-size dataset for a large number of epochs to converge, which requires powerful hardware resources and long pre-training time. According to those limitations, I choose to use the ELECTRA model to approach the research question, as it is a smaller, yet powerful, descendant of BERT.

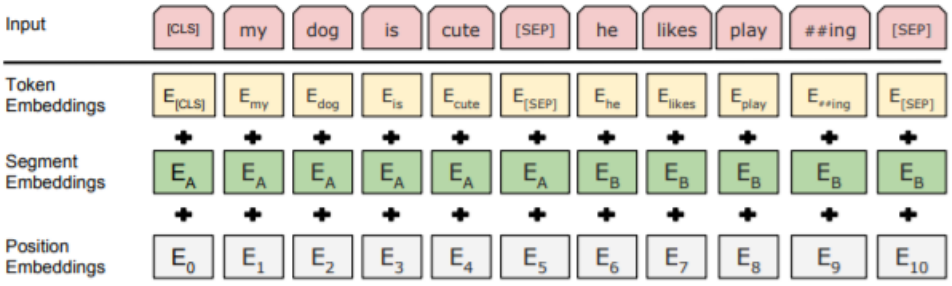


Figure 2.1: BERT Input Representation, Adapted from BERT (Devlin *et al.*, 2019). It shows the three embeddings layers for an example sequence which consist of two sentences with the [SEP] special token in between.

2.2 ELECTRA

The Efficiently Learning an Encoder that Classifies Token Replacements Accurately (ELECTRA) model was introduced by Clark *et al.* (2020) with a novel pre-training task that made the model outperform the previous state-of-art BERT model. The pre-training task used by the ELECTRA model is the Replaced Token Detection (RDT) task. This task solves the drawbacks of the MLM task used in the BERT model, such as: all input tokens contribute to the loss calculation for the model vs. only 15% of them within the MLM task, and it solves the inconsistency of the appearance of the masked token [MASK] in pre-training task but not in fine-tuning downstream tasks. Consequently, these modifications yield higher performance than the BERT model with the same size or even larger-size models. These results were reported using the GLUE (Wang *et al.*, 2019) and the SQuAD (Rajpurkar *et al.*, 2016a) metrics.

In the RTD task, the masked tokens [MASK] are replaced with randomly sampled tokens from the input, then the model is trained to predict if each token is real or fake. Hence, the ELECTRA model neither needs a massive size of text data for

pre-training (as it makes use of all the input tokens, not just 15% of them), nor a large number of training steps. Moreover, the NSP task, which is used in the BERT model pre-training, is not used in the ELECTRA model pre-training, as the portion of the performance improvement resulting from this task is unremarkable (Devlin *et al.*, 2019).

The input representation to the ELECTRA model is structured the same way as in the BERT model. However, the model architecture is different, as the architecture for the ELECTRA model consists of Generator and Discriminator models. The Generator is a trained MLM model, which masks the input tokens randomly and predicts the correct identity of that token from a pool of possible tokens. On the other hand, the Discriminator is trained by replacing the masked input tokens with other tokens sampled by the generator, and the Discriminator model is trained to predict if each token within the input sequence is true or fake. The ELECTRA model's architecture and an example of the RTD task is illustrated in Figure 2.2.

Since the ELECTRA-base model outperforms the BERT-base model, a smaller size was presented in the ELECTRA paper called ELECTRA-small. The way they downsized the model is by reducing the embedding layer's size; from 256 to 128, the number of hidden layers; from 768 to 256 and the batch size from 256 to 128, while increasing the learning rate, the number of training steps and the max_seq_length parameter; from 512 to 128. The size variations of ELECTRA-large, ELECTRA-base and ELECTRA-small are summarized in Table 2.1. Compared to the BERT model, ELECTRA is smaller in size, uses less data for pre-training and outperforms BERT for all GLUE language understanding fine-tuning tasks, which is the main reason I choose to work with ELECTRA for my approach.

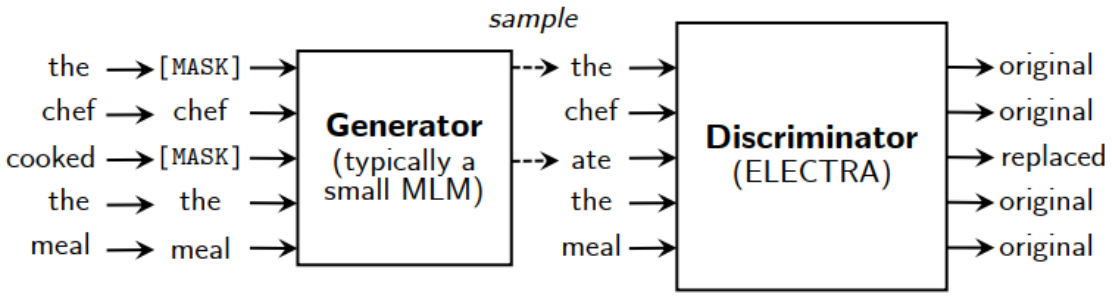


Figure 2.2: The Generator-Discriminator architecture of the ELECTRA model, with an example of the RDT task in pre-training. This figure is Adapted from (Clark *et al.*, 2020)

Hyperparameter	Small	Base	Large
Number of layers	12	12	24
Hidden Size	256	768	1024
Attention heads	4	12	16
Attention head size	64	64	64
Embedding Size	128	768	1024
Generator Size	1/4	1/3	1/4
Mask percent	15	15	25
Learning Rate Decay	Linear	Linear	Linear
Warmup steps	10000	10000	10000
Batch Size	128	256	2048
Learning Rate	5e-4	2e-4	2e-4
Train Steps (ELECTRA)	1.45M/1M	1M/766K	464K/400K

Table 2.1: Variations of the ELECTRA model with different model sizes. The parameters are reported in the ELECTRA paper(Clark *et al.*, 2020). I use this table as a reference to reduce the ELECTRA model size, in order to adapt it to work with the small data size.

2.3 Words-As-Classifiers (WAC)

As opposed to text-based Language Models like ELECTRA and BERT, which can only learn the textual semantics of words, the Words-As-Classifiers (WAC) model (Schlangen *et al.*, 2016) can capture the concrete meanings of words. The WAC

model is a lightweight visual semantics model that is used to create a classifier for each word from the input text. In contrast to language models, it is trained on visual features extracted from images. From the definition, the dataset needed to train the WAC model consists of both words and images that co-occur with these words (i.e., referring expressions made to objects in the images). Furthermore, the trained word classifiers encode the visual meaning of words. The parameters of the word classifiers can be used as visual word-level embeddings.

2.3.1 Model Architecture

The WAC model consists of two main sub-models. The first one is used to extract visual features from images, and it follows a transfer-learning scheme; the model is pre-trained on a visual dataset (images) and further can be adapted to be used on a similar task using a different visual dataset. An example of this is the CLIP ViT-B/32 model (Radford *et al.*, 2021), which is a conventional neural network pre-trained on images from the ImageNet dataset (Russakovsky *et al.*, 2015). The CLIP model extracts visual features from an image creating a feature vector of size 512. The second sub-model is a binary classification model that uses the visual features vectors —extracted from input images using the CLIP model— as input to train a binary classifier for each word from the input text data. This classifier can be a Logistic Regression, a Multi-Layer Perceptron (MLP) or a Decision Tree model.

2.3.2 Data

As previously mentioned, the data used to train the WAC model consists of both images and text. The input text is tokenized into words, and the images are converted to visual feature vectors. Furthermore, those vectors are used as positive and negative examples to train each word's WAC classifier. The Image-Text data used to train the

WAC model can be "coupled" or "non-coupled" data. The coupled data consists of images and text that are associated together from the same source, for example: images with text descriptions, images with text captions and images with referring expressions. On the other hand, the text and images of the non-coupled data are collected from different sources, for example: collecting images for certain words using Google search; those images are gathered from different sources. As for the work done in this thesis, both the RefCOCO and VisDial datasets are adapted as sources of coupled Image-Text data, as illustrated in sections 2.5 and 2.6.

2.3.3 Training WAC

The procedure to train the WAC model follows a number of steps. The first step is text tokenization, where each sentence from the input text is tokenized into a set of words. The next one is the visual features extraction step, in which all the input images are transformed into vectors. In this step, the visual features are extracted from images using the pre-trained CLIP ViT-B/32 model that transforms the image into a feature vector of size 512. Hereafter, a classifier model for each word is trained using the previously-created feature vectors; those vectors work as positive and negative examples based on whether an image (represented by a feature vector) is associated with a word or not. For example, the word "zebra" occurs in referring expressions to images that include a zebra, so those images are used as positive examples of the word "zebra". However, all other images are considered negative examples for the same word "zebra", as the word "zebra" did not appear to describe those images. The binary classification model I use is the logistic regression with parameters (random_state= 123, penalty=l2, C= 1000). Finally, I extract the coefficients of the trained classifiers for words as vectors of size 513, which captures

the visual meaning of each word. Additionally, I concatenate it with another vector of size 7, that holds information about the relative position of images [1]. The final vector size is 520, which I use as a visual word-level embeddings for words.

2.3.4 Drawbacks

Although the WAC model captures the multi-modal meaning of words (visual meaning) as opposed to the text-based LMs, it does not capture syntactic or textual semantics for words. Hence, a word's meaning is independent of the context or its position in a sentence.

2.4 ELECTRA + WAC

To overcome some of the transformer-based language model limitations and extend their knowledge to multi-modal representations, Kennington (2021) presented a new model that combines the ELECTRA model with the WAC model to form a visual-textual semantics model, that can be pre-trained on textual and visual data altogether. This model uses the WAC model to create visual embeddings for a set of words, and the ELECTRA model to extract the contextual embeddings from the input text. In the following sub-sections, I explain the model and its pre-training regime.

2.4.1 Model Architecture

The key element in this model is the embedding layer of the conventional ELECTRA model, which is the layer that ties the Generator and Discriminator models,

[1] The positional vector can only be calculated in case of using the RefCOCO dataset, as the text is referring to only an object within the image (sub-image), which has x-y dimensions and area. However, the text in the VisDial dataset refers to the objects within the whole image not just a single object within the image, and consequently, there is no sub-image to calculate its relative position and area within the original image (The vector is zeros). I highlight this as a difference/limitation between the RefCOCO and VisDial dataset in Section 2.6.

where they share the words (tokens) embeddings. The ELECTRA+WAC model replaces the word embeddings with the WAC visual-semantics embeddings while pre-training the Discriminator model and freezing the embedding layer weights for a number of pre-training epochs. Hence, the model is able to learn both the visual and textual embeddings together. The architecture of this model consists of the ELECTRA-small model with the same settings shown in Table 2.1. The parameters of the WAC model used are: the CLIP's ViT-B32 model as the visual feature extraction model, and the logistic regression model (C=0.25, and max_iterations =1000) as the binary classifier. The model's architecture is illustrated in Figure 2.3.

2.4.2 Data

Distinctly, the data used to train this model is a combination of text and images. The text data is the OpenWebText dataset (Gokaslan & Cohen, 2019), which is a corpus-based dataset of size 40GB (the same data used to pre-train the ELECTRA model (Clark *et al.*, 2020)). Alternatively, the data used to train the WAC classifiers is a non-coupled data of 27,152 words adapted from the BERT Vocabs list, with 100 images per word, collected using an automated Google search. However, the remaining words from the BERT Vocabs list [2]have a visual word embeddings vector of zeros.

2.4.3 Model Pre-Training

The first step in pre-training the ELECTRA+WAC model is to fully train the WAC model to create a classifier for each word (of the 27,152 words). Subsequently, pre-training an ELECTRA-small model while swapping the contextual word embeddings (generated by ELECTRA) with the WAC embeddings for the same words,

[2]The BERT Vocabs list has approximately 30K words.

and "Freezing" the weights of the embeddings layer. Hence, the model does not overpower the WAC embeddings, giving the model a chance to learn and maintain the visual embeddings driven by the WAC model. The freezing operation is a technique applied during pre-training the ELECTRA model, by enforcing the embeddings layer to have fixed word embeddings and relying on that while pre-training the Generator and Discriminator models, which enables the ELECTRA model to learn the injected visual embeddings without the contextual embeddings overpowering them.

2.4.4 Evaluation

The authors of the ELECTRA+WAC model reported their results using the MRPC, the CoLA and the WNLI tasks from the GLUE benchmark. Both the CoLA and WNLI scores matched the ELECTRA-small model scores. However, the MRPC score was slightly increased by 0.062 and 0.024 for F1 and accuracy respectively.

These results state that the new ELECTRA+WAC model can learn visual-textual semantics, and when fine-tuned on pure language understanding tasks (like the GLUE tasks), its performance is equivalent or slightly higher than the ELECTRA-small model (trained only on text). The results from this work imply that using visual representations to train abstract language models, can enhance — or at least equalize— the model performance when measured using pure language understanding tasks (i.e., GLUE tasks). Furthermore, I am inspired to test the current hypothesis using this model, to explore the outcome of using the proposed approach in enhancing the abstract language models and its effect on pure language understanding tasks (i.e., GLUE tasks).

2.4.5 Drawbacks

Although the ELECTRA+WAC model learns the syntactic representations of the

English language, it utilizes the WAC visual embeddings for 27,152 words of the BERT Vocabs to pre-train the ELECTRA-small language model, which is a rough assumption that all the words have concrete meanings (all the words have WAC visual embeddings). Moreover, this assumption is as extreme as the one implied by text-based LMS, like the ELECTRA-small model, that all words are abstract (all word meanings are learned solely from text). Nonetheless, the human language is rich in both concrete and abstract meanings, and adding this distinction to LMs would —according to my hypothesis— enrich LMs and narrow the gap between machine and human understanding of the human language. Another drawback of this model is the usage of non-coupled data to train the WAC model. Additionally, the images used to extract the visual meaning of words are collected using an automated Google search, as opposed to other datasets that are collected using human subjects. I overcome those drawbacks using the methodology proposed in the next chapter.

2.5 RefCOCO Dataset

In order to address the research question, a dataset that is representative of the way children acquire language is used, as according to McCune (2008), children first acquire language via referring expressions; such as the text in the RefCOCO dataset. The RefCOCO consists of 19,994 images from Microsoft COCO (MS-COCO) images (Lin *et al.*, 2014), that includes 50,000 objects with 142,209 referring expressions describing these objects. The RefCOCO dataset was created by using the MS-COCO dataset and the ReferitGame (Kazemzadeh *et al.*, 2014) to add human-generated referring expressions to the objects within images. In this two-player game, the first player is given an image with a segmented target object and is asked to write an expression referring to this object. On the contrary, the second player only has access

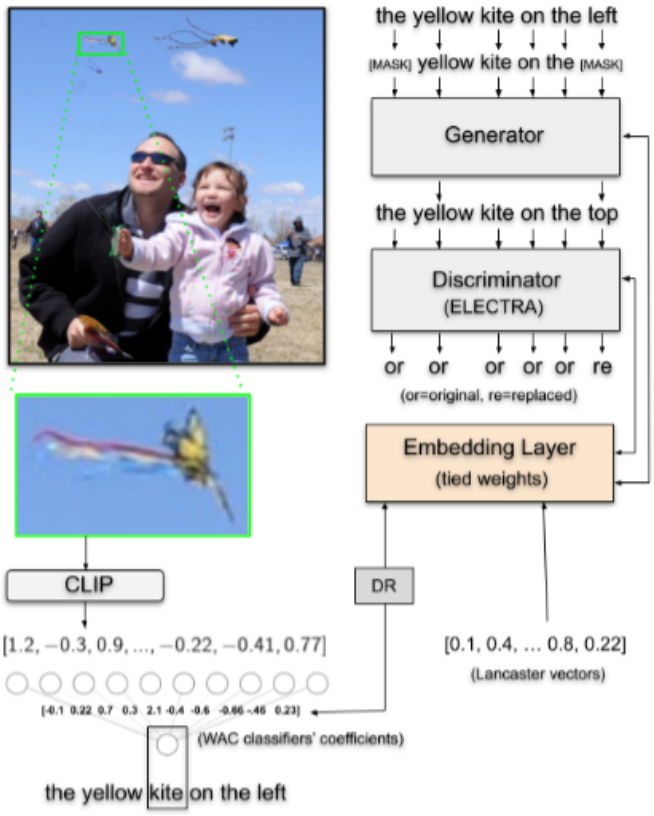


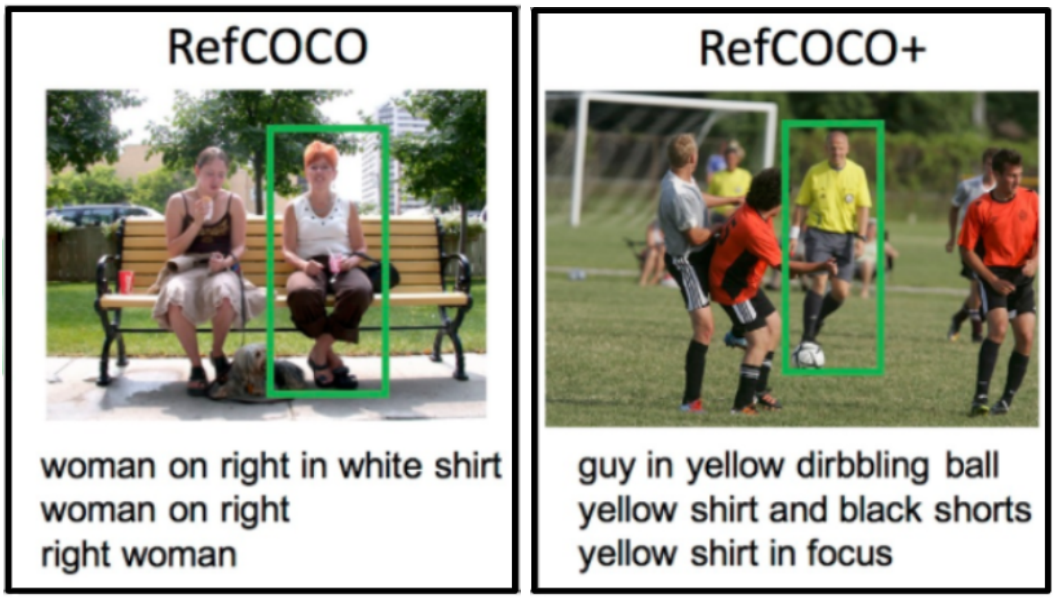
Figure 2.3: A visual illustration of the ELECTRA+WAC model, adapted from (Kennington, 2021). In this figure, a feature vector is extracted from the kite object using the CLIP pre-trained model. Next, a classifier for the word "kite" is created, and concatenated with the Lancaster norms vector for the same word. Finally, the WAC embeddings vector becomes the representation of the word "kite", and ELECTRA–small is pre-trained using this embeddings vector.

to the image and referring expressions, and is asked to choose the object corresponding to them. If the players do their job correctly, they receive points and swap roles, otherwise, they start a new turn. In the RefCOCO dataset, there are no restrictions on the type of language used in the referring expressions, which can have positional words such as "right" and "left" [3]. The RefCOCO+ is another variation of this

[3]The positional words are dealt with in the WAC model by adding positional information to the visual features, as a vector of 7 digits; this information includes the relative x-y coordinates of the

dataset that does not allow any positional information in referring expressions and mainly concentrates on the appearance-based description. I use the RefCOCO dataset while incorporating the positional information with the WAC embeddings. Examples for the two datasets are shown in Figure 2.4a and Figure 2.4b. The RefCOCO dataset has its Train, Validation and Test splits for ease of development. The length of referring expressions tends to be short (in terms of words). For example, the average length of referring expressions extracted from the RefCOCO (Split= Train) dataset is 3.4 words, which is different from the average length of sentences from a corpus dataset (as the OpenWebText (Gokaslan & Cohen, 2019)). Another difference between the two types of text data is the size, as the size of RefCOCO (Split= Train) and OpenWebText is 2.6MB and 40GB, respectively.

visual objects within the image, and the area of the object.



(a) An Example of RefCOCO dataset. The referring expressions refer to the woman in white as the object or sub-image of the original image. It is noted that this dataset has positional words as "right".

(b) An Example from RefCOCO+ dataset. The referring expressions refer to an object within the original image, which is the man in yellow short. This dataset does not have positional words.

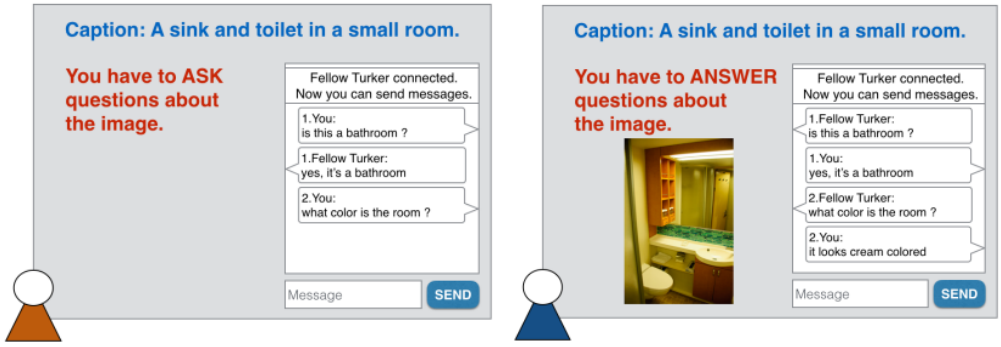
Figure 2.4: The RefCOCO and RefCOCO+ dataset examples, adapted from (Yu *et al.*, 2016).

2.6 VisDial Dataset

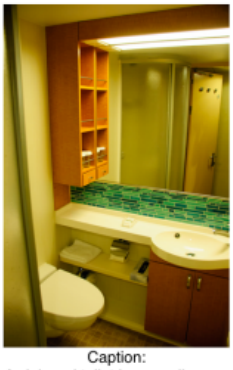
Due to the small text size of the RefCOCO, a similar dataset is added to leverage the total size of the text data used in this work. The VisDial dataset (Das *et al.*, 2017) is visually-grounded dialog data based on images from the MS-COCO dataset. This VisDial v0.9 dataset consists of approximately 1.2M dialogs based on 120k images, and each dialog consists of 10 rounds of questions/answers. This dataset is collected using two main human individuals: the "Questioner" and "Answerer", both share the caption of the image but only the "Answerer" can see the image. The game

starts when the "Questioner" asks a question — with a total of 10 questions — to the "Answerer" to identify the components/features of the image, as the dialog between them describes the whole image's content. An example of this approach is depicted in Figure 2.5. The dataset also has pre-defined data splits (Train and Validation) to ease the development. I only use the answers portion of dialogs from this dataset with the exception of the yes/no answers as it does not hold any visual information nor contextual information (single word sentence). The average answer length in this dataset is 2.9 words as reported in (Das *et al.*, 2017), which is again far smaller than a normal corpus dataset (like OpenWebText).

Although adding this dataset to the RefCOCO dataset increases the data size needed by the data-hungry transformer LM, there are some differences between both datasets, such as: first, in the RefCOCO dataset the referring expressions describe a single object from the image, while in the VisDial dataset, dialog reflects information about the whole image (with multiple objects). The second difference is that the answers from the VisDial dataset are not necessarily referring to expressions. However, they describe objects within the source images.



(a) A "Questioner" Interface. **(b) An "Answerer" Interface.**



Caption:
A sink and toilet in a small room.

Q3: can you see anything else ?
A3: there is a shelf with items on it
Q4: is anyone in the room ?
A4: nobody is in the room
Q5: can you see on the outside ?
A5: no, it is only inside
Q6: what color is the sink ?
A6: the sink is white
Q7: is the room clean ?
A7: it is very clean
Q8: is the toilet facing the sink ?
A8: yes the toilet is facing the sink
Q9: can you see a door ?
A9: yes, I can see the door
Q10 what color is the door ?
A10 the door is tan colored

(c) A dialogue example in VisDial dataset.

Figure 2.5: The VisDial data collection procedure example, adapted from (Das *et al.*, 2017).

2.7 GLUE

The goal of this thesis is to evaluate the effect of the proposed training regime (explained in chapter "Method") on enhancing the traditional language models, by using language understanding tasks. The General Language Understanding Evaluation (GLUE) (Wang *et al.*, 2019) benchmark is widely used to evaluate models on a collection of sentences/sentence-pair NLU tasks, using pre-defined metrics for each task. The GLUE benchmark has nine NLU tasks; each task has its train and test

datasets that vary in size, and each task is measured by a pre-defined metric. Those tasks belongs to one of three categories: single-sentence, similarity and paraphrasing and natural language inference. All tasks and metrics are summarized in Table 2.2. These tasks are all classification tasks except for the STS-B which is a regression task. All nine tasks are illustrated next.

CoLA

The Corpus of Linguistic Acceptability (CoLA), this task is a single-sentence classification task; the data consists of sentences of annotated acceptable/non-acceptable grammatical English examples, that are extracted from books and journals. The result is reported using Mathew's correlation.

SST-2

The Stanford Sentiment Treebank (SST-2) is a binary sentiment inferences (positive/negative) task, where the sentences are extracted from movie reviews annotated by human subjects. The data is balanced and the result is reported with the accuracy metric.

MRPC

In the Microsoft Paraphrase Corpus (MRPC), the data consists of a pair of automatically-extracted sentences that is annotated by human subjects, whether this pair of sentences have the same meaning or not. This data is imbalanced so it is measured using both accuracy and F1 score metrics.

QQP

The Quora Question Pairs dataset (QQP) consists of question pairs collected from the Quora community, and the task is to determine if a pair is semantically equivalent or not. This is also imbalanced data; hence, it is reported using accuracy and F1 score metrics.

STS-B

The Semantic Textual Similarity Benchmark (STS-B) consists of sentence pairs that are collected from different sources, such as: news headlines, video captions and image captions. This is a regression task to report the similarity of each pair with a score that ranges from 0.0 to 5.0, with 5 as high similarity and 0 as no similarity.

MNLI

The Multi-Genre Natural Language Inference corpus (MNLI) (Williams *et al.*, 2018) is a crowd-sourced collection of sentence pairs with textual entailment annotations. This data consists of a premise and a hypothesis sentence pairs, and the task is to predict if the premise entails the hypothesis, contradicts the hypothesis, or neither (neutral). The premise sentences are gathered from ten different sources, including transcribed speech, fiction, and government reports.

QNLI

The Question Answering Language Inference (QNLI) task is based on the Stanford Question Answering dataset (Rajpurkar *et al.*, 2016b). This dataset consists of question-paragraph pairs, where one of the sentences in the paragraph contains

the answer to the corresponding question (written by an annotator). However, it is converted to question-sentence pairs where sentences are extracted from the corresponding paragraph. The task is to determine whether the context sentence contains the answer to the question or not.

RTE

The Recognizing Textual Entailment (RTE) dataset is extracted from a series of annual textual entailment challenges, where the data from RTE1, RTE2, and RTE3 are combined to form the final dataset. The task is an entailment task for two sentences, and the result is reported using the accuracy metric.

WNLI

The Winograd Natural Language Inference task is based on the Winograd Schema Challenge (Levesque *et al.*, 2012), which is a reading comprehension challenge in which a model's task is to read a sentence with a pronoun and select the referent of that pronoun from a list of choices. This task reports the accuracy metric.

2.8 Related Work

The high performance of the BERT model led many scholars to study and direct their research to solve its limitations, such as: the need for a large size of data, the need for highly-efficient hardware resources, and the large pre-training time needed by the model to converge, not to mention the fact that it does not capture any physical meanings for words. This resulted in several approaches developed by scholars to solve one or more of these limitations for better and more feasible language models.

Corpus	Train	Test	Task	Metrics	Domain
Single-Sentence Tasks					
CoLA	8.5k	1k	acceptability	Matthews corr.	misc.
SST-2	67k	1.8k	sentiment	acc.	movie reviews
Similarity and Paraphrase Tasks					
MRPC	3.7k	1.7k	paraphrase	acc./F1	news
STS-B	7k	1.4k	sentence similarity	Pearson/Spearman corr.	misc.
QQP	364k	391k	paraphrase	acc./F1	social QA
Inference Tasks					
MNLI	393k	20k	NLI	matched/mismatched acc.	misc.
QNLI	105k	5.4k	QA/NLI	acc.	Wikipedia
RTE	2.5k	3k	NLI	acc.	news, Wikipedia
WNLI	634	146	coreference/NLI	acc.	fiction books

Table 2.2: A summary of the GLUE language understanding tasks. The tasks are divided by the task category into single-sentence tasks, similarity and paraphrase tasks, and Inference tasks. The table shows the different sizes of training and evaluation datasets for each task. The table also illustrates the source of the dataset and metrics used to measure each task. The highlighted metrics are the ones I report in my evaluation.

Some of these contributions attempt to optimize BERT's size by minimizing its parameters, such as the TinyBERT model (Jiao *et al.*, 2020), which uses the knowledge distillation technique to transfer knowledge from a larger to a smaller model referred to as "Teacher" and "Student" models, respectively. In this contribution, the BERT model is used as a teacher to create the TinyBERT model. The distillation is used in both the pre-training and fine-tuning steps of the new model. This model's performance is outperformed by the ELECTRA model, measured using the GLUE benchmark. Other contributions were developed to automatically optimize the model as in the auto TinyBERT model (Yin *et al.*, 2021). In the context of enabling LMs to be multi-modal, other contributions worked towards combining text with visual semantics of words to form a holistic knowledge of language semantics. Those models can be used in tasks such as: visual dialog, visual question answering (VQR) and

text-image retrieval. Some of these models are based on the BERT model with minor changes in architecture to accommodate the addition of the visual data, such as the VilBERT model (Lu *et al.*, 2019) which is a two-stream model for text and vision with a co-attention layer connecting both at the end. This model is pre-trained on the conceptual captions dataset (Sharma *et al.*, 2018) of 3.3 million images. However, this dataset represents a weakly-connected image to text data where the text is automatically collected from the HTML "alt-text" for images. The drawbacks of this model are: the text and visual features are learned independently, the size of the model exceeds the BERT model size, and the architecture combines a pre-trained BERT model for text features with a CNN-based model (ResNet) for extracting the visual features of the images. Another BERT-based model is the LXMERT model (Tan & Bansal, 2019), which is also based on the transformer's architecture. The VilBERT model consists of three encoder models for the multi-modality, one for the visual object features, another for the language semantics and the third one is for combining both. This model implements extra pre-training tasks (other than the MLM and NSP) to accommodate the addition of visual input (images). The drawback of this model is that it is a large-size model that is trained on five aggregated data sources (data hungry). Another BERT descendant model is the PixelBERT (Huang *et al.*, 2020), which is an end-to-end model trained with both visual and sentence semantics to map image pixels to text. The model has two encoder models to manipulate visual and textual data, and their outputs are concatenated to form the final representation. The vision encoder is based on a CNN backbone that randomly samples pixels from the whole image and removes the urge to use a dataset with bounding boxes for objects. Both the visual genome and MS-COCO dataset (101K

and 106K Images, respectively) were used to pre-train this model, and it is pre-trained on both the MLM and Image-Text Matching (ITM) tasks. Another model (Xu *et al.*, 2021) Implemented the encoder/decoder architecture as End-to-End vision language pre-training is (reference) which used MS-COCO and the same pixel-level features extraction as pixelBERT without the random pixel masking that the latter is trained through, this model can be fine-tuned to more tasks such as object detection and caption generation. The training of such a model is very time-consuming as visual and textual pre-training gets pre-trained together and the total loss is dependent on all of its components.

Other contributions used visual semantics to specifically improve visual dialog tasks by using multi-modal dialog models as the work done by Murahari *et al.* (2020), which extended the VilBERT model's architecture and the general pre-training dataset (Conceptual Captions) with a task-related dataset (VQA dataset), then it was fine-tuned on Visual dialog datasets. Additionally, a recent contribution in this field is the Maria model (Liang *et al.*, 2021), which has a dialog-specific architecture that extends BERT with a parallel visual transformer model to extract text and visual features. This model also requires huge data as it is trained on both MS-COCO and OpenImages datasets.

All of those contributions rely on using a massive amount of image-text data, where the text portion is a rich corpus and they all made the assumption that all the words are both abstract and concrete to the same degree and their meanings are learned from both sources together, which I overcome with my proposed approach.

CHAPTER 3:

METHOD

3.1 Overview

In order to approach the research question, I extend the ELECTRA+WAC model, as it is a light model that makes use of both visual and contextual embeddings. However, due to the text data limitations (previously discussed), I make some modifications to the model's parameters (without changing the architecture). I also apply other modifications in the pre-training regime to incorporate adding concreteness knowledge to this model. I explain the details of the data pre-processing step, the modifications applied to the model's architecture and the procedures applied to cover all aspects of the research question.

3.2 Data Pre-processing

One of the essential parts of this work is choosing the data that will help me explore my hypothesis. Recall that the hypothesis implies imitating the approach children learn their first language when exposed to the physical world. I break down the hypothesis into the following factors: using visual representations of the world, using the input text as referring expressions (as it is how children first learn the language), and differentiating between abstract and concrete words, with each learned from the corresponding source (concrete words are learned from the visual data and abstract

words are learned from the contextual data). Having all of this in mind, I use data that would best help me in illustrating my ideas and performing experiments with minimal limitations. The data I use can be divided into two sub-categories that have two different roles: the first is the referring expression data, which consists of simple sentences that are accompanied by a physical scene defining them, for example: "horse with pink girl riding on it", "girl on a skateboard", and "chocolate sprinkle donut". The second sub-category of data I use is to provide a mean of distinction between concrete vs. abstract words. This type of data associates a numeric score to words; this score defines how concrete/abstract is a word (from the perspective of human subjects). For example: according to the dataset, the word "blue" has a higher concreteness score than the word 'bravery", which means that it is recognized as more concrete. This type of data I further refer to as concreteness distinction knowledge data. In this section I introduce the data I use, the limitations of the data and the pre-processing steps applied.

3.2.1 Referring Expressions Image-Text dataset

The first type of data I use is the referring expressions that refer to objects in a visual scenery. For this purpose, I use both the RefCOCO (Schlangen *et al.*, 2016) and Visdial (Das *et al.*, 2017) datasets. Both of those datasets have text that describes images, and the images are adapted from the MS-COCO dataset (Lin *et al.*, 2014), and both datasets have pre-defined splits of the data (Train, Validation and Test). This type of data is used in two ways: the text portion of the data is used as input text to pre-train the ELECTRA model and get textual embeddings for words. Moreover, the Image-Text coupled data is used to train the WAC model and extract visual embeddings for words that build up the referring expressions.

Throughout experiments, I split the text portion of the data into three sizes that grow ascendingly (measured in Bytes), which are Size1; has text from RefCOCO (Split = Train), Size2; has text from RefCOCO(Split = TrainTestValidation)[1] and Size3; has text from Size2 combined with text from VisDial(Split= TrainValidation)[2]. As shown in Table 3.1, Size1 consists of 120,624 sentences with a size of 2.14 MB, and Size2 consists of 142,210 sentences of size 2.54 MB, whereas Size3 consists of 1,018,950 sentences of size 16.9 MB. Notably, all three sizes are comparably small to the textual data size used by the ELECTRA model[3]. The reason I use different sizes in the current approach is to study the effect of the pre-training data size on the model performance and to enhance the reliability of the results, as I expect to notice a rule that interprets my results across different data sizes.

From the Refcoco dataset, I extract the referring expressions that describe different objects within a visual context (an image). Moreover, the VisDial dataset is processed by extracting the answers and separating them with a newline (same as with the RefCOCO dataset). Answers with only "Yes" and "No" are omitted because the minimum sentence length for an example used in pre-training ELECTRA is two (no context to learn from).

Although both datasets are based upon the same visual source (Images from MS-COCO dataset), the text extracted from both datasets is slightly different. The text in RefCOCO refers to a single object in an image that has multiple objects. Consequently, the text refers to a sub-image of the original image, whereas in the

[1]The RefCOCO(Split = TrainTestValidation) is the size of the text in the Train, Test and Validation splits combined, which are all the textual referring expressions within the RefCOCO dataset.

[2]The VisDial(Split = TrainValidation) is the size of the text in the Train and Validation splits combined.

[3]ELECTRA is trained on 40 GB of the OpenWebText dataset.

VisDial dataset, the dialogue corresponds to the whole image. This difference affects the subsequent extraction of visual embeddings using the WAC model. The other difference results from the way both datasets were created. Although both are created from human interaction, the settings were different and for different tasks, which is illustrated in Sections 2.5 and 2.6.

Data	Size1	Size2	Size3
Data Size(Sentences)	120,624	142,210	1,018,950
Data Size(Bytes)	2.14 MB	2.54 MB	16.9 MB
Mean Sentence Length(words)	3.496	3.505	4.08
Words with WAC Visual Embeddings	9,350	10,328	11,265

.

Table 3.1: The different data sizes used through this work.

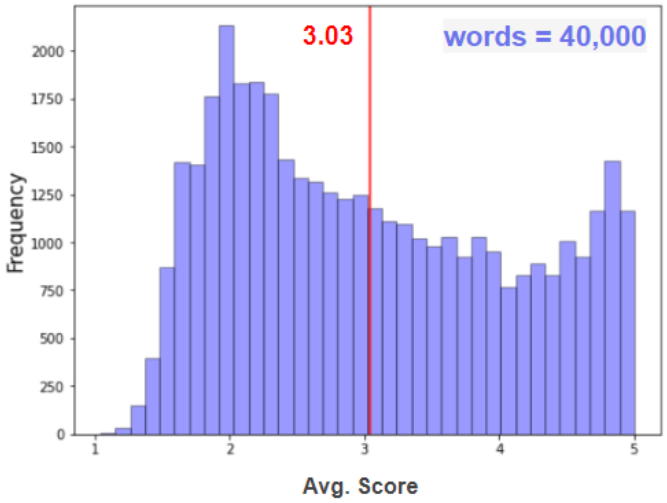
3.2.2 Concreteness Distinction Data

The second type of data utilized in this work is the concreteness distinction data, which allows the model to identify concrete/abstract words. Consequently, the model does not assume that all words are concrete as in the WAC and the ELECTRA+WAC models, nor all are abstract as in ELECTRA models. In addition, through the following procedures, I set multiple threshold values for concreteness, with each threshold value defining a different list of concrete words. These threshold values range from the assumption that all words are concrete to the assumption that all words are abstract. Furthermore, a model corresponding to each threshold value is pre-trained. For this type of data, two different, yet correlated, datasets are used. These datasets are The Concreteness Score dataset (Brysbaert *et al.*, 2014) and the Age of Acquisition dataset (Kuperman *et al.*, 2012).

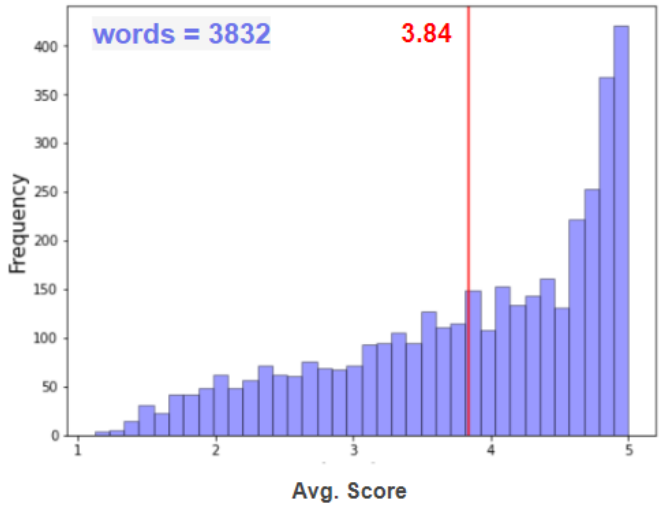
The Concreteness Score Dataset

The Concreteness Score dataset (Brysbaert *et al.*, 2014) has 40,000 words; each word is given multiple scores from different human subjects (28 people). As shown in Figure 3.1, the average score (decimal number) per word ranges from 1 to 5, with 1 being abstract and 5 being concrete. Figure 3.2a depicts the distribution of the average score per word for this dataset. I set concreteness threshold values for this dataset from 1 to 5 with a step of 1.

Figure 3.1: The concreteness score values and their concreteness level. The color purple represents abstract, and the salmon color is for concrete. For analysis, less concrete notation is used rather than more abstract; thus, the 3 threshold's color is light salmon.



(a) Distribution of the average score for words in the Concreteness Score dataset.



(b) Distribution of the intersection of words from the Concreteness Score dataset and RefCOCO (split=Train).

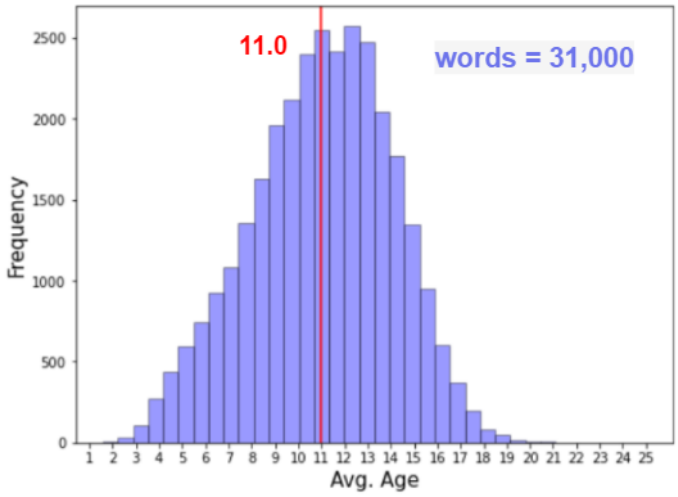
Figure 3.2: The sub-plots present the distribution for the avg. score per word in the Concreteness Score dataset. Sub-plot(a) shows the distribution for the original dataset, while sub-plot(b) shows the distribution for the intersection of the Concreteness Score dataset with RefCOCO (Split=Train). This distribution is clearly more concrete as words with avg. concreteness score of 4 and 5 are more than words of 1 and 2, and the mean slightly increase. However, the intersection excludes the majority of words from the Concreteness Score dataset, as the total number of words is reduced from 40,000 to 3832 words.

The Age of Acquisition (AoA) Dataset

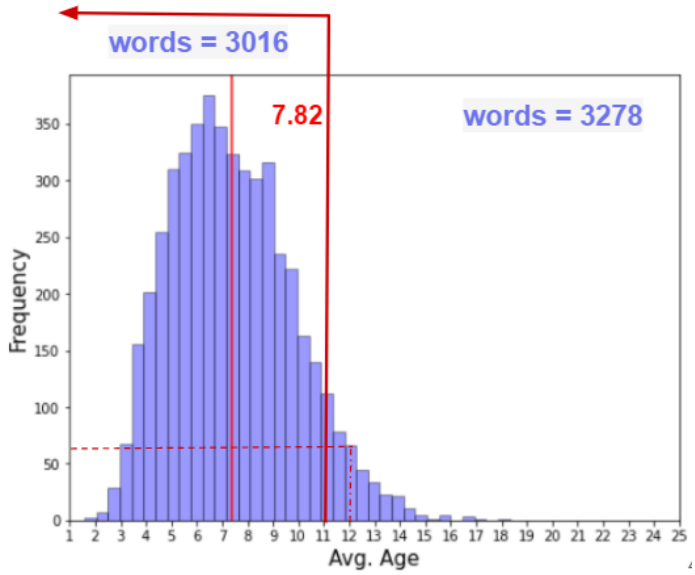
The AoA dataset (Kuperman *et al.*, 2012) is used to provide the concreteness distinction knowledge. This dataset contains average ratings for the age of acquisition of over 30,000 English words, approximating the average age at which these words are learned, ranging from 1.58 to 25 years. The distribution for the ages is depicted in Figure 3.4a. Figure 3.5 demonstrates the positive correlation between the AoA and Concreteness Score datasets, which serves as evidence that the final results are not random. Borghi *et al.* (2019) have demonstrated that children learn concrete words before abstract words, which means that words learned at younger ages tend to be concrete, whereas words learned at older ages tend to be abstract. Hence, I make the assumption that this dataset can be used as an indication of the concreteness of a word.

The procedures implemented in this thesis use various threshold values, from age=2 to age=11 with a step of 1, representing the transition from concrete to abstract ages, respectively, as demonstrated in Figure 3.3. This subset of the ages is selected because, according to the correlation in Figure 3.5, the age=11 corresponds to a concreteness score below 3 (leaning towards abstract), and age=2 corresponds to a concreteness score of 4 (concrete), as well as the gray area of concreteness knowledge (between concrete abstract points). Furthermore, in Figure 3.4b, the intersection of the AoA and the RefCOCO dataset denotes that ages over 11 and below 2 have minor occurrences in the RefCOCO referring expressions, which confirms the previous insight.

Figure 3.3: Age values from the AoA dataset and their concreteness level. The color purple represents abstract, and the salmon color is for concrete. In this analysis, less concrete notation is used rather than more abstract notation. Thus, the 8,9 and 10 threshold's color is light salmon (not light purple).



(a) Distribution for avg. age of acquisition for words in the AoA dataset.



(b) Distribution for the avg. age of acquisition for words that intersects with words in the RefCOCO (split=Train) dataset.

Figure 3.4: Estimating the concreteness ages from the AoA dataset.

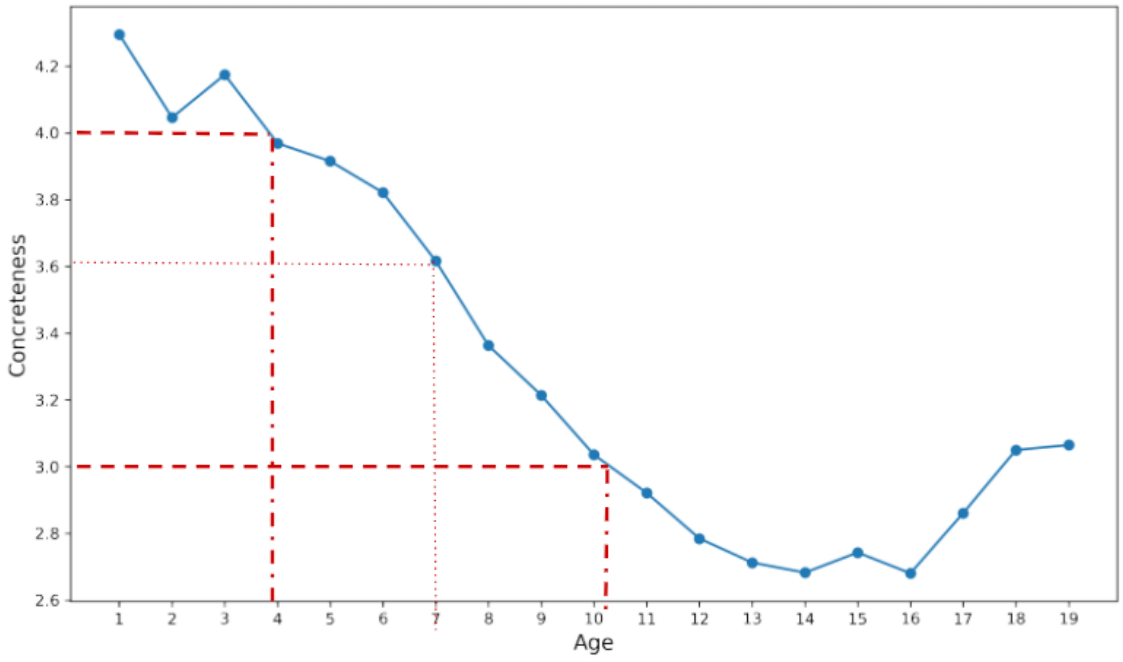


Figure 3.5: Correlation between the AoA dataset and the Concreteness Score dataset.

3.2.3 Concreteness Thresholds

Multiple concreteness thresholds are derived from both the AoA and the concreteness score datasets in order to examine the effect of adding the concreteness distinction knowledge to model pre-training. These values vary between the two extreme assumptions made by the ELECTRA and the WAC models, which assume that all words have only contextual meanings (all words are abstract) and all the words have visual meaning (all words are concrete), respectively.

Based on the threshold value and the source dataset (AoA or Concreteness Score), a substantial subset of words are distinguished as concrete, whereas all other words are abstract. This subset of concrete words varies according to different threshold values. For the AoA dataset, concreteness decreases with increasing age; therefore,

for a word to be concrete for a given threshold value, its average age must be less than or equal to the threshold value. However, for the concreteness Score dataset, a word is concrete for a particular threshold value if the average score for this word is more than or equal to the threshold value. Using those previous rules, the model can distinguish concrete words; hence, all other words are abstract. Furthermore, this knowledge is used to select the appropriate source of embeddings for each word within the text dataset. For concrete words, the embeddings are extracted from the WAC embeddings, and the embeddings for the abstract words are extracted from contextual ELECTRA embeddings. The result is mixed word embeddings derived from the two textual and visual sources for a specific threshold value. This mixing step is provided for all threshold values (15 threshold values from both datasets), which leads to 15 different combinations of mixed word embeddings.

It is worth noting that words distinguished as concrete for some threshold values are misleading. For instance, a word with an average score of 2.5 in the concreteness score dataset, for a threshold value of 2, is considered concrete, while the 2.5 score is not concrete according to the dataset definition. In order to eliminate this confusion, three categories are set for the threshold values: The Al_ Abstract, the All_Concrete, and the True_Concrete category. The All_abstract category groups threshold values for which all the words are assumed to be abstract, which means that the set of concrete words — according to the threshold value— is empty (no or few words are concrete). On the contrary, the All_concrete category groups together all threshold values that assume all words are concrete (no or few words are abstract). The final category is the True_Concrete category, which has threshold values that distinguish actual concrete words ——according to humans' perspective—— as concrete. For

example: according to the concreteness score dataset, if the threshold value is 4, then all words with an average score equal to or more than 4 are considered concrete, which aligns with the assumption True_Concrete. However, a threshold value of 1 from the same concreteness dataset is considered All_Concrete, as it assumes all the words with an average score equal to or more than 1 are concrete; which is all the words in the dataset. On the contrary, the threshold of 5 is an All_Abstract threshold, as no words in the dataset have an average score of more than 5, so all the words are abstract. The threshold values and categories are demonstrated in Figure 3.7 and Figure 3.6.

According to the current hypothesis, the performance of models that corresponds to one or multiple thresholds, leaning toward the True_Concrete threshold category, is expected to experience significantly higher values than other models of All_Abstract and All_Concrete threshold categories. These threshold values represent the best values where the model can learn visual embeddings from visual sources (WAC) and contextual embeddings from text-based sources (ELECTRA-Xsmall). Furthermore, models are given names which reflect the threshold value that provided the concreteness distinction knowledge to their pre-training. For example: a model pre-trained using the concreteness distinction knowledge of threshold= 2 from the AoA dataset is called "mix_AoA_2". Consequently, these models inherit the same concreteness category as the corresponding thresholds, therefore, the model mix_concreteness_5 is considered an All_Abstract model (as threshold 5 of the concreteness score dataset is an All_Abstract threshold).

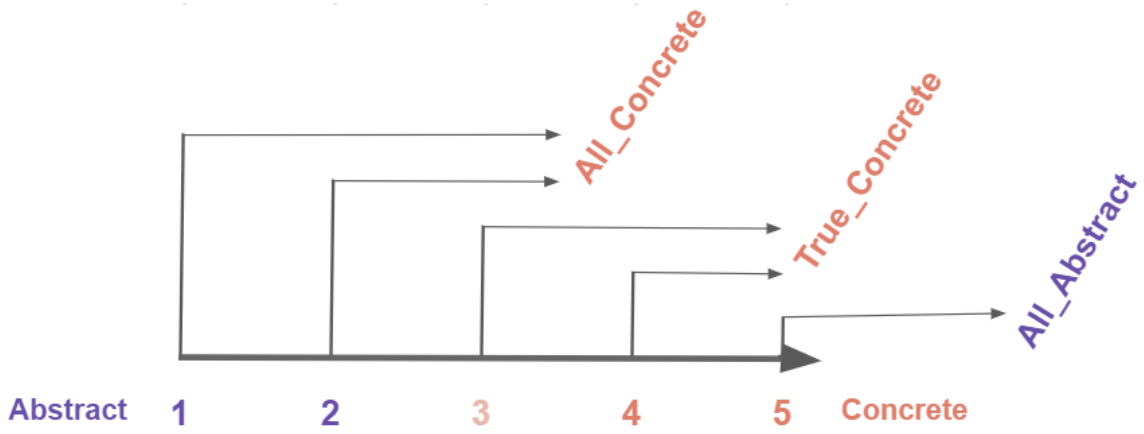


Figure 3.6: The different thresholds from the concreteness Score dataset, Each threshold value belongs to one of the threshold categories; All_Abstract, All_Concrete, and True_Concrete. The All_Concrete thresholds are the ones that have the largest set of words (they assume approximately all the words are concrete.)

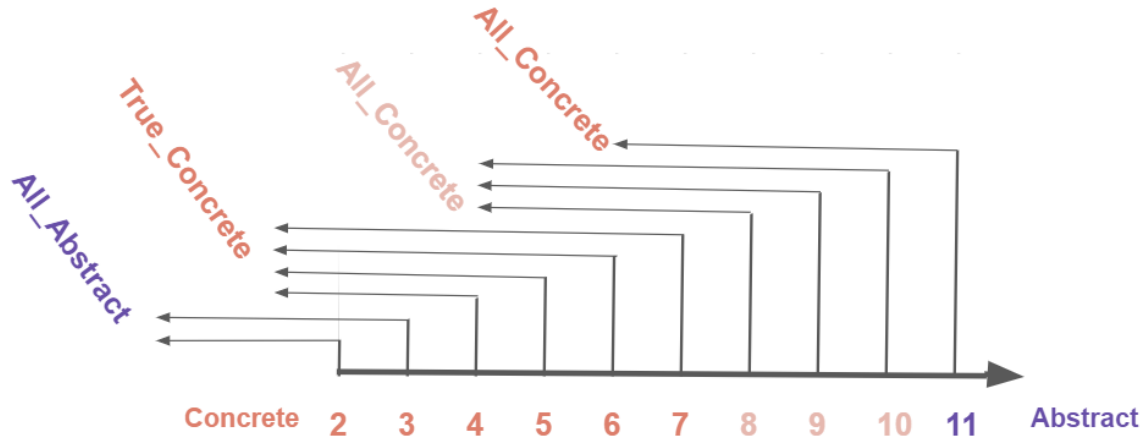


Figure 3.7: The different thresholds from the AoA dataset, each threshold value belongs to one of the threshold categories; All_Abstract, All_Concrete, and True_Concrete. The All_Concrete threshold are the ones that have the largest set of words (they assume approximately all the words are concrete).

Data Size	Threshold	Concreteness Score Words	WAC Words	Intersection with WAC Words	
				Number	Percentage
Size1	mix_concreteness_1	39,738	9,350	3832	40.98%
	mix_concreteness_2	31,714		3608	38.59%
	mix_concreteness_3	18,308		3004	32.13%
	mix_concreteness_4	8996		1983	21.21%
	mix_concreteness_5	0		0	0.00%
Size2	mix_concreteness_1	39,738	10,328	4059	39.30%
	mix_concreteness_2	31,714		3814	36.93%
	mix_concreteness_3	18,308		3172	30.71%
	mix_concreteness_4	8996		2096	20.29%
	mix_concreteness_5	0		0	0.00%
Size3	mix_concreteness_1	39,738	11,265	5758	51.11%
	mix_concreteness_2	31,714		5315	47.18%
	mix_concreteness_3	18,308		4239	37.63%
	mix_concreteness_4	8996		2704	24.00%
	mix_concreteness_5	0		0	0.00%

Table 3.2: The number of words for each threshold value from the Concreteness Score dataset, the number of words that has WAC visual embeddings for each data size and the intersection — in number and percentage — between the Concreteness Score dataset and the words which have WAC visual embeddings, for all the three sizes of data (Size1, Size2 and Size3).

Data Size	Threshold	AoA Dataset Words	WAC Words	Intersection with WAC Words	
				Number	Percentage
Size1	mix_AoA_2	3.00	9,350	2.00	0.02%
	mix_AoA_3	48.00		42.00	0.45%
	mix_AoA_4	338.00		277.00	2.96%
	mix_AoA_5	994.00		724.00	7.74%
	mix_AoA_6	2,043.00		1,266.00	13.54%
	mix_AoA_7	3,497.00		1,800.00	19.25%
	mix_AoA_8	5,423.00		2,230.00	23.85%
	mix_AoA_9	8,067.00		2,598.00	27.79%
	mix_AoA_10	11,267.00		2,868.00	30.67%
	mix_AoA_11	14,956.00		3,016.00	32.26%
Size2	mix_AoA_2	3.00	10,328	2.00	0.02%
	mix_AoA_3	48.00		42.00	0.41%
	mix_AoA_4	338.00		280.00	2.71%
	mix_AoA_5	994.00		741.00	7.17%
	mix_AoA_6	2,043.00		1,309.00	12.67%
	mix_AoA_7	3,497.00		1,867.00	18.08%
	mix_AoA_8	5,423.00		2,325.00	22.51%
	mix_AoA_9	8,067.00		2,722.00	26.36%
	mix_AoA_10	11,267.00		3,018.00	29.22%
	mix_AoA_11	14,956.00		3,189.00	30.88%

Size3	mix_AoA_2	3.00	11,265	2.00	0.02%
	mix_AoA_3	48.00		43.00	0.38%
	mix_AoA_4	338.00		303.00	2.69%
	mix_AoA_5	994.00		834.00	7.40%
	mix_AoA_6	2,043.00		1,550.00	13.76%
	mix_AoA_7	3,497.00		2,328.00	20.67%
	mix_AoA_8	5,423.00		3,019.00	26.80%
	mix_AoA_9	8,067.00		3,672.00	32.60%
	mix_AoA_10	11,267.00		4,159.00	36.92%
	mix_AoA_11	14,956.00		4,471.00	39.69%

Table 3.3: The number of words for each threshold value from the AoA dataset, the number of words that have WAC visual embeddings for each data size, and the intersection — in number and percentage — between the AoA dataset and the words which have WAC visual embeddings, for all the three sizes of data (Size1, Size2, and Size3).

3.3 Model Architecture

The architecture used in this work is based on the architecture used in (Kennington, 2021) and illustrated in Figure 2.3. However, the architecture is extended and modified to incorporate the small size of text data used and the addition of concreteness knowledge of the physical world to the model. The developed architecture consists of a smaller version of the transformer-based model ELECTRA-small; which I call ELECTRA-Xsmall, the WAC model to extract the visual embeddings for words within the data, and the third part is a "Multiplexer" [4], which is responsible for

[4]The Multiplexer term is borrowed from logic circuits design, it is a logic circuit where the output

choosing the embedding source to be either visual or textual based on whether the word is considered concrete or abstract, determined by the concreteness threshold value (as previously explained). The overall architecture is illustrated in this section, and the input to each component is visually demonstrated in Figure 3.8.

3.3.1 WAC

The WAC model is one of the main components used in this work to create visual embeddings for words using coupled Image-Text datasets (i.e., RefCOCO and VisDial). Furthermore, these visual embeddings are assigned only to words distinguished as concrete (by the concreteness threshold). Originally, the WAC embeddings vectors are of size 520, which I shrink to size 128 (the size of ELECTRA-small's embedding layer) using the Umap model (McInnes *et al.*, 2018).

Some of the original WAC parameters are modified, such as the minimum number of positive examples for a word to be considered to have a classifier model is changed from 3 —as used in the previous literature— to 1, due to the small number of words that overcome this constrain in the used dataset (approximately 3000 words in RefCOCO (Split=Train)). This number will be further decreased when intersected with the list of words in the concreteness distinction knowledge datasets. Thus, the previous change is applied to Size1 and Size2 of the data and left as it is for Size3, as it contains a large number of words, precisely 37,297 unique words, and with the constrain, it is limited to 11,265 words. The total number of WAC word-level embeddings is 9,350, 10,328, and 11,265 words for Size1, Size2, and Size3, respectively.

---

is equal to only one of the inputs, based on the value of another signal independent from the input. In this work, I mimic the functionality of a Multiplexer to choose the source of the embeddings for words in input text, determined by the threshold value.

3.3.2 ELECTRA-Xsmall

In order to incorporate the small data used through my experiments, the size of ELECTRA-small is reduced into an even smaller version that I call ELECTRA-Xsmall. The ELECTRA-small model is trained on OpenWebText data of size 40 GB, while the text data from both RefCOCO and VisDial is only 16.9 MB. This significant reduction in data size prevented the model from completing pre-training using fewer examples for 100K epochs. This size reduction is implemented by following some of the approaches used in the ELECTRA paper to shrink ELECTRA-large to ELECTRA-base and ELECTRA-small. Specifically, the batch size and max_seq_length parameters are reduced to get the model to complete pre-training for 100K epochs on Size1 (2.14 MB) of the text data. The new parameters are: batch size = 10 and the max_seq_length = 8[5]. All the other model parameters are the same as ELECTRA-small, as depicted in Table 3.4.

3.3.3 Model Pre-Training

The ELECTRA-Xsmall model follows the pre-training and fine-tuning approach same as the ELECTRA-small model. However, in the proposed architecture, depicted in Figure 3.8, the ELECTRA-Xsmall + WAC is used with the addition of concreteness knowledge to the model during pre-training.

The pre-training process goes through the following steps: First, train the WAC classifiers for words in the input text (i.e., Referring Expressions), using the images from the same dataset. Second, pre-train the ELECTRA-Xsmall model using the exact input text (Referring Expressions), and extract the contextual embeddings for

[5]Sentences with sequence_length less than the max_seq_length are padded with zeros (same as in ELECTRA-small model), while the ones with sequence_length more than the max_seq_length are truncated.

Hyperparameter	Electra-small	Electra-Xsmall
Number of layers	12	12
Hidden Size	256	256
Attention heads	4	4
Attention head size	64	64
Embedding Size	128	128
Generator Size	1/4	1/4
Mask percent	15	15
Warmup steps	10K	10K
Batch Size	128	10
max_seq_length	128	12
Train Steps (ELECTRA)	1.45M/1M	100K

Table 3.4: The change in parameters for the ELECTRA-Xsmall model from the original ELECTRA-small model.

the words learned by the fully trained ELECTRA-Xsmall model. At this point, there are two different sources for word embeddings, and its up to the concreteness knowledge to decide which words are concrete hence their embeddings shall be the WAC visual embeddings, while abstract words get assigned the contextual embeddings, which is referred to as mixing the embeddings, and this is the third step in the pre-training process. This step I apply for all concreteness threshold (15 thresholds) values. Finally, the ELECTRA-Xsmall is pre-trained using the corresponding mixed embeddings for each threshold value. These new embeddings are embedded into the embedding layer between the Generator and Discriminator in ELECTRA-Xsmall, using the freezing approach followed by Kennington (2021) and illustrated in section 2.4.

Despite the model pre-training consisting of four simple steps, the current experiments have lots of variations due to the multiple parameters I use to address different parts of the current research question. These parameters include: first, the data size that is either Size1, Size2, or Size3, which are all child-development-inspired datasets

but of different sizes (as summarized in Table 3.1). Second, the data source is either the proposed dataset (child-development inspired dataset) or a corpus dataset that is typically used to train language models (i.e., OpenWebText dataset). Although using the conventional corpus dataset is not part of the research question, it was included in the experiments to provide a baseline to compare against the proposed model with the sole difference of the data source. The third parameter is the concreteness threshold value, which are 15 different threshold values, in addition to a single conventional ELECTRA-Xsmall model pre-trained without adding any concreteness knowledge. These variations result in running 96 models (3 text data sizes, 2 text data sources, and 15 concreteness threshold values, and a conventional ELECTRA-Xsmall model). An automated pipeline is created with different input parameters for each set of experiments, and this procedure is illustrated in the next section.

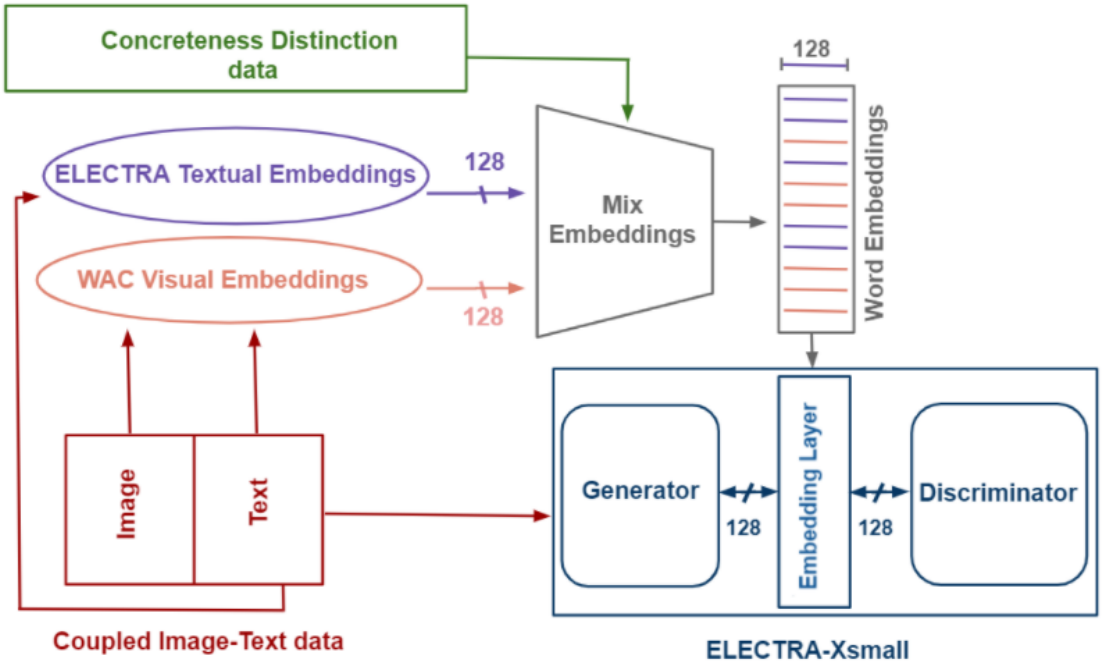


Figure 3.8: Model architecture using coupled text-images dataset to train the WAC model for visual representations and ELECTRA-xsmall model for textual representation. The decision upon which embeddings vector per word is decided based on the concreteness threshold extracted from the concreteness distinction knowledge datasets.

3.4 Procedures

According to the research hypotheses presented in this work, the effect of following a child-development inspired approach is investigated to pre-train language models, and it is hypothesized to improve the performance of LMs compared to the traditional pre-training techniques that absolutely lack knowledge about the physical world and are trained using the large, heavily-structured textual corpus dataset. Consequently, a language model is pre-trained using child-development inspired dataset and with knowledge about the physical world (visual embeddings for words) and the distinction between concrete vs. abstract words. After pre-training the model, its performance

is compared with the same model pre-trained on a different dataset (corpus), without the addition of the concreteness distinction knowledge.

Three procedures are formulated as shown in Table 3.5. The first two procedures share the same settings except for the concreteness distinction knowledge source, as procedure1 is applied using the Concreteness Score dataset and procedure2 is performed using the AoA dataset. For each concreteness threshold value, an ELECTRA-Xsmall model is pre-trained with mixed word embeddings that correspond to the concreteness threshold value. This step is repeated for each data size (Size1, Size2, and Size3). Since both the AoA and Concreteness Score datasets positively correlate together, the first and second procedures are considered as a mean of affirmation to each other. Unlike the first two procedures, the input text used in the third one is a corpus dataset (i.e., OpenWebText) as opposed to the child-development inspired dataset, used in them. Additionally, the third procedure is carried out using all concreteness thresholds from datasets of concreteness distinction knowledge while using a corpus dataset with a comparable size for the three data sizes (Size1, Size2, and Size3).

3.4.1 Procedure1

For this procedure, an ELECTRA-Xsmall is pre-trained on the three data sizes, using the visual embeddings from the WAC model trained on the referring expressions from the corresponding coupled Image-Text data for this size. Overall, 6 models per size are created; hence, it ends up with 18 different models, each corresponding to 18 different (data size, concreteness score threshold/no threshold) combinations, which is performed according to the method mentioned in Section 3.3.3.

3.4.2 Procedure2

This procedure is a replica of the first procedure, except for the source of concreteness distinction knowledge is rived from the AoA dataset. For this procedure, ten different ELECTRA-Xsmall models are pre-trained, each corresponding to a concreteness threshold, which is repeated for different data sizes (Size1, Size2, Size3), and thus 30 models are obtained for the (data size, AoA threshold) combinations, as illustrated in Table 3.5.

3.4.3 Procedure3

This procedure is follows the pattern used in the two preceding procedures; the only difference is the type of input text data used to train the ELECTRA-Xsmall model. Instead of the referring expressions used in the first two procedures, this one uses corpus text data from the OpenWebText (Zhu *et al.*, 2015) dataset. However, other parameters remain the same as the other procedures; the concreteness thresholds and their corresponding WAC embeddings and the mixed embeddings are identical to the two preceding procedures. Moreover, not all the OpenWebText data is used to train these models, as the goal of this procedure is to enrich the analysis by providing a baseline to compare results from the first two procedures against; hence, three fragments of text data are extracted from the OpenWebText dataset with sizes comparable to Size1, Size2 and Size3 of the referring expressions text data (measured in bytes), as shown in Table 3.1. Consequently, the number of models obtained from this procedure is 48 for the (data size, 16 AoA and Concreteness Score thresholds/no threshold) combinations. It should be noted that the image and text data are considered non-coupled Image-Text datasets, as the images come from one source (RefCOCO and VisDial), and the text is from another(OpenWebText).

Procedures		**Procedure1**	**Procedure2**	**Procedure3**	
Dataset Used		Referring Expressions	Referring Expressions	**Corpus**	
Dataset Size	**Size1**	RefCOCO (Split=Train)	RefCOCO (Split=Train)	Comparable size of RefCOCO (Split=Train)	
	Size2	RefCOCO (Split=TrainValTest)	RefCOCO (Split=TrainValTest)	Comparable size of RefCOCO (Split=TrainValTest)	
	Size3	RefCOCO (Split=TrainValTest) + VisDial (Split=TrainVal)	RefCOCO (Split=TrainValTest) + VisDial (Split=TrainVal)	Comparable size of RefCOCO (Split=TrainValTest) + VisDial (Split=TrainVal)	
Concreteness Knowledge		Concreteness Score	AoA	Concreteness Score	AoA
Concreteness Threshold values		[1,5] With a step of 1	[2,11] With a step of 1	[1,5] With a step of 1	[2,11] With a step of 1
Visual Embeddings		WAC Trained on Dataset Sizes (Size1, Size2, Size3) of coupled Text-Images	WAC Trained on Dataset Sizes (Size1, Size2, Size3) of coupled Text-Images	WAC Trained on Dataset Sizes (Size1, Size2, Size3) of coupled Text-Images	
Textual Embeddings		ELECTRA Xsmall Trained on Datasets size (Size1, Size2, Size3) of Text	ELECTRA Xsmall Trained on Datasets size (Size1, Size2, Size3) of Text	ELECTRA Xsmall Trained on Datasets of comparable size (Size1, Size2, Size3) of Text	

Table 3.5: The procedures performed to produce all the 96 different models, each model corresponding to a different threshold value, text data source, and text data size. This includes 6 conventionally trained ELECTRA–Xsmall models for each data size and data source (3*2), without any visual embeddings. Procedure1 and Procedure2 both use the pre-training text data as referring expressions, and Procedure3 uses the corpus dataset with comparable sizes to Size1, Size2, and Size3 (measured in bytes).

CHAPTER 4:

EVALUATION

For the purpose of evaluating my work, I split the research question into sub-questions, and I address each one of them independently through analysis, while highlighting the limitations to which my analysis is subjected. After that, I summarize the results with a reflection that supports/denies my hypothesis. In the next subsections I introduce the metrics I use and the evaluation sub-tasks, in addition to the results I collected from my experiments altogether.

4.1 Metrics

Due to the nature of the underlying research question and the baselines against which it is compared, determining the appropriate metric to evaluate the model has been difficult. Recall the research question, "Can I move towards emulating the settings in which children learn their language to enrich language models? On a more technical level, if language models are given knowledge of the physical world, with a distinction between concrete and abstract words, and a dataset that is similar to the child-development inspired dataset, resulting in better model performance. The main objective is to determine whether LMs will benefit from this new approach. Hence, the GLUE benchmark, which measures the performance of LMs on various language understanding tasks, is used as the metric to evaluate my work. Despite

the fact that the current model is multi-modal, I need to determine whether adding such features to the model would leverage its "understanding" when applied to pure language tasks. The second reason I choose the GLUE benchmark is the variety of tasks/data sources, as each task in GLUE has a different dataset that comes from different data sources and varies in size.

Some of the GLUE tasks are excluded from the evaluation for different reasons. For example, according to what is hypothesized by Devlin *et al.* (2019) that the higher the model size, the less fine-tuning data it needs; thus, GLUE tasks with small training datasets would not be informative to my work due to the limitation of the small pre-training data size I encountered that consequently needs a smaller model. This limitation excludes the CoLA, MRPC, and STS-B tasks from the evaluation tasks I use. This is supported by results from previous work by Kennington (2021), where the CoLA metric is characterized by very low accuracy (approaching zero) accuracy using the WAC embeddings with the freezing approach in pre-training, with the pre-training data is the whole OpenWebText of 40GB. In contrast, the SST-2 and MNLI metrics will best reveal the strength of my approach, as the nature of the SST-2 dataset (movie reviews) and the fact that it is a single-sentence task makes it the best metric for overcoming many drawbacks of my work. In addition, the MNLI has a massive dataset, which will overcome the small pre-training dataset. Additionally, the WNLI and RTE metrics are added to the list of fine-tuning tasks used, to provide further analysis of results. The GLUE tasks I use are highlighted in Table 2.2. In contrast to the approach followed in the BERT and the ELECTRA papers, I use no hyper-parameter tuning for the GLUE tasks; consequently, all tasks use the same fine-tuning parameters, including batch size, number of epochs, and

learning rate, as shown in Table 4.1.

It is worth noting that the GLUE script used in my work is not the same as the one described in ELECTRA (Clark *et al.*, 2020) and ELECTRA+WAC (Kennington, 2021) papers. This GLUE script is adapted from the ELECTRA Pytorch implementation[1], and it does not use the built-in ELECTRA tokenizer from the Huggingface library. This script provides flexibility to adjust the BERT Vocabs easily —add new words from each text data size— and use it with consequent pre-training/fine-tuning steps. However, to ensure the validity of this script, I compare the results reported for the ELECTRA-small model in (Clark *et al.*, 2020) with the pre-trained ELECTRA-small model (Published by Google, through the HuggingFace library) and fine-tuned using this GLUE script. Both models report comparably equal results. On the other hand, to ensure the validity of the ELECTRA-small Pytorch implementation used, it is fine-tuned on the GLUE tasks and compared to results from the pre-trained ELECTRA-small model (Published by Google, through the HuggingFace library). Only a slight difference is found between the two models, which can be attributed to the difference in the number of training steps from 100K to 1000K between the two models, respectively. All of those results are highlighted in Table 4.2.

[1]From Github: https://github.com/lucidrains/electra-pytorch

Parameter	Value
max_seq_length	128
per_gpu_train_batch_size	32
learning_rate	2e-5
weight_decay	0.0
adam_epsilon	1e-8
max_grad_norm	1.0
num_train_epochs	3

Table 4.1: A summary of the parameters used to fine-tune the GLUE tasks. Unlike the approach used in previous literature (Devlin *et al.*, 2019) and (Clark *et al.*, 2020). I perform no hyper-parameter tuning on any GLUE tasks.

4.2 Baselines and Other Points of Comparison

4.2.1 General Baselines

Although variant experiments are applied to approach the research question, the ELECTRA-Xsmall model of the corresponding data size is used as the baseline for all of them, as it represents the abstract LM, not only independent of adding the visual representations but also independent of freezing the embeddings layer weights of the model. Moreover, models with different settings are evaluated against each other in my analysis to investigate the effect of the multiple factors within the current approach.

4.2.2 Downsizing ELECTRA-small

The effect of downsizing the ELECTRA model is considered one of the present

work's most significant limitations. The process of selecting adequate model parameters is substantial to this study; thus, I report the GLUE tasks metrics for the ELECTRA model with the different batch_size and max_sequence_length parameters, starting from ELECTRA-small and progressing to the model ELECTRA-Xsmall. The dataset used to pre-train models in this analysis is the OpenWebText dataset. The results of this experiment are reported using the CoLA, MNLI, MRPC, SST-B, STS-2, RTE and WNLI tasks. As expected, reducing the size of the model has a significant negative impact, and using a smaller model size than ELECTRA-small yields lower results for all GLUE tasks used. The most obvious ones were CoLA, MRPC, and STS-B tasks. As both CoLA and MRPC tasks report no change in results, with CoLA reporting 0 accuracy and MRPC as 0.748 accuracy, while the STS-B reports an extreme decay in correlation from 0.78 to 0.08 going from ELECTRA-small to ELECTRA_64_128[2], and with going lower, NaN values (due to division by a zero value when calculating Spearman's correlation). There is no significant difference in the MNLI accuracy between the smaller sizes, but it decreased significantly from 0.79 to 0.33 between the ELECTRA-small and the ELECTRA_64_128 sizes, which is a substantial change. The RTE has less decay going from 0.608 to 0.527. Those results are summarized in Table 4.2.

[2]The ELECTRA_64_128 model has a batch_size=64 and max_seq_length=128, while the ELECTRA-small model has a batch_size=128 and max_seq_length=128.

Model_name	SST-2	CoLA	MRPC	STS-B	MNLI	RTE	WNLI
ELECTRA_10_12 (Xsmall-ours)	0.6910	0.0000	0.7480	NaN	0.3545	0.5307	0.5634
ELECTRA_32_128	0.5092	0.0000	0.7480	NaN	0.3182	0.4729	0.5634
ELECTRA_64_128	0.5275	0.0000	0.7480	0.0842	0.3223	0.5271	0.5634
ELECTRA _128_128 (small-ours)	0.8727	0.4703	0.8508	0.7796	0.7926	0.6173	0.5352
ELECTRA-small (Google)	0.9105	0.5550	0.8678	0.8610	0.8175	0.6209	0.5211
ELECTRA-small (Google)*	0.8910	0.5460	0.8370	0.8300	0.7970	0.6080	-

Table 4.2: A demonstration of the effect of downsizing the ELECTRA-small model using GLUE tasks [SST-2, MRPC, STS-B, MNLI, RTE, and WNLI]. The models are trained with smaller batch_size and max_sequence_length parameters, on the OpenWebText dataset. The last three models are all ELECTRA-small, trained by using ELECTRA-small Pytorch implementation. The second is trained and published by Google through the Huggingface library (Wolf *et al.*, 2019). Both those models are evaluated using the open-source GLUE script used in my analysis. The last one is the model reported by Google in the ELECTRA paper (Clark *et al.*, 2020); the WNLI metric is not reported in this paper. It is clear that using ELECTRA models smaller than the ELECTRA-small reduces the model's performance.

4.3 Experiments

4.3.1 The Effect of Adding Visual Embeddings

In order to address this sub-question, models that had the visual embeddings added during their pre-training are compared with models that did not/almost did not have a visual representation in their pre-training. In other words, models that represent the True_Concrete threshold values are compared with others with the All_Abstract thresholds and with the ELECTRA-Xsmall model. Furthermore, those models are compared in terms of the data size to determine if adding the visual embeddings to pre-training would outperform models without them and pre-trained

on a larger data size. Furthermore, this effect is investigated by using corpus data and referring expressions data independently. The results compared are reported in two sets of four tables, with each table reporting one GLUE task. The results for True_Concrete models are reported in tables: 4.3, 4.4, 4.5 and 4.6 , and the All_Abstract models are reported in tables: 4.7, 4.8, 4.9 and 4.10.

Using Corpus Data

Across The Same Data Size

For SST-2, MNLI, RTE, and WNLI tasks, no difference in accuracy between True_Concrete and All_Abstract models is observed. However, the RTE task demonstrated a slight decrease in accuracy for the All_Abstract models [mix concreteness_5 and mix_AoA_3] to be 0.47 and the same accuracy for the True_Concrete models [mix_AoA_4 and mix_AoA_5], which might be because these models have close thresholds values.

Across Different Data Sizes

Comparing the results from True_Concrete to models with less pre-training data from All_Abstracts revealed no change in performance.

Using Referring Expressions Data

Across The Same Data Size

For SST-2 and MNLI tasks, no change is noticed between the All_Concrete and All_Abstract model pre-trained on Size1 and Size3. However, for Size2 a change in the max accuracy between the two threshold categories is noticed. I find that All_Abstract is outperforming the True_Concrete thresholds with max accuracy of

0.70 and 0.55 respectively. On the other hand, the MNLI task, regarding Size2 of data, has a change in the opposite direction, as the True_Concrete is outperforming the All_Abstract thresholds model, with max accuracy of 0.50 (reported by model mix_AoA_5) and 0.43 respectively. The RTE and WNLI tasks report no change in accuracy.

Across Different Data Sizes

Comparing the results from True_Concrete to models with less pre-training data from All_Abstracts revealed no change in performance.

Discussion

Based on previous results, the addition of visual embeddings did not improve the performance of the corpus-trained models, independent of the data size. This is also noticed for models pre-trained on referring expressions for Size1 and Size3 of the data. However, All_Abstract models perform 0.2 higher accuracy than the All_Concrete models, which implies that adding the visual knowledge to abstract models —using the methodology applied in this work— does not improve the model performance on language understanding tasks.

Limitations

The limitation that applies in this instance is the low percentage of concrete words that intersect with words of the referring expressions and corpus data, as determined by each threshold value. This percentage is reflected in Tables 3.3 and 3.2. Another limitation is the intersection between corpus data and words with visual embeddings, as some corpus words may not have visual embeddings from the WAC model, which is a data inconsistency.

Data_Size	Model_Name	Corpus	Referring Expressions
Size1	ELECTRA-Xsmall	0.509174	0.509174
	mix_AoA_2	0.509174	0.509174
	mix_AoA_3	0.509174	0.509174
	mix_concreteness_5	0.509174	0.497706
Size2	ELECTRA-Xsmall	0.509174	0.707569
	mix_AoA_2	0.509174	0.509174
	mix_AoA_3	0.509174	0.600917
	mix_concreteness_5	0.509174	0.707569
Size3	ELECTRA-Xsmall	0.509174	0.813073
	mix_AoA_2	0.509174	0.805046
	mix_AoA_3	0.509174	0.814220
	mix_concreteness_5	0.509174	0.816514

Table 4.3: The SST-2 results for using referring expressions vs. corpus textual data, while pre-training using models with All Abstract thresholds and the ELECTRA-Xsmall model as well, which have all the same settings except for freezing the word embeddings layer feature.

Data_Size	Model_Name	Corpus	Referring Expressions
Size1	ELECTRA-Xsmall	0.318186	0.318186
	mix_AoA_2	0.318186	0.318186
	mix_AoA_3	0.318186	0.318186
	mix_concreteness_5	0.318186	0.318186
Size2	ELECTRA-Xsmall	0.318186	0.439531
	mix_AoA_2	0.318186	0.318186
	mix_AoA_3	0.318186	0.353439
	mix_concreteness_5	0.318186	0.439531
Size3	ELECTRA-Xsmall	0.318186	0.585430
	mix_AoA_2	0.318186	0.586449
	mix_AoA_3	0.318186	0.571065
	mix_concreteness_5	0.318186	0.573917

Table 4.4: The MNLI results for using referring expressions vs. corpus textual data, while pre-training using models with All Abstract thresholds and the ELECTRA-Xsmall model as well, which have all the same settings except for freezing the word embeddings layer feature.

Data_Size	Model_Name	Corpus	Referring Expressions
Size1	**ELECTRA-Xsmall**	0.527076	0.527076
	mix_AoA_2	0.527076	0.527076
	mix_AoA_3	0.527076	0.527076
	mix_concreteness_5	0.527076	0.527076
Size2	**ELECTRA-Xsmall**	0.527076	0.527076
	mix_AoA_2	0.527076	0.527076
	mix_AoA_3	0.527076	0.530686
	mix_concreteness_5	0.527076	0.527076
Size3	**ELECTRA-Xsmall**	0.523466	0.545126
	mix_AoA_2	0.527076	0.530686
	mix_AoA_3	0.472924	0.527076
	mix_concreteness_5	0.472924	0.498195

Table 4.5: The RTE results for using referring expressions vs. corpus textual data, while pre-training using models with All Abstract thresholds and the ELECTRA-Xsmall model as well, which have all the same settings except for freezing the word embeddings layer feature.

Data_Size	Model_Name	Corpus	Referring Expressions
Size1	**ELECTRA-Xsmall**	0.43662	0.56338
	mix_AoA_2	0.56338	0.56338
	mix_AoA_3	0.56338	0.56338
	mix_concreteness_5	0.43662	0.56338
Size2	**ELECTRA-Xsmall**	0.43662	0.56338
	mix_AoA_2	0.43662	0.56338
	mix_AoA_3	0.43662	0.56338
	mix_concreteness_5	0.43662	0.56338
Size3	**ELECTRA-Xsmall**	0.56338	0.563380
	mix_AoA_2	0.56338	0.394366
	mix_AoA_3	0.56338	0.563380
	mix_concreteness_5	0.56338	0.577465

Table 4.6: The WNLI results for using referring expressions vs. corpus textual data, while pre-training using models with All Abstract thresholds and the ELECTRA-Xsmall model as well, which have all the same settings except for freezing the word embeddings layer feature.

4.3.2 The Effect of using Child-Development Inspired dataset

The first characteristic that distinguishes it from corpus datasets is that it is comprised of simple short sentences as opposed to complex lengthy sentences in corpus data. The second characteristic of this data is that it is considered to emulate children's interaction with the physical world in terms of describing physical objects from a visual scene. Hence, each feature is approached independently to answer this question. For the first sub-question, the effect of using different textual input

sources for pre-training the model, so All_Abstract thresholds are used to perform the analysis independent of the effect of using visual embeddings. In this setting, the results between models with different data sources are used for the same size and across different sizes of the data. On the contrary, the second sub-question needs the addition of visual embeddings. Consequently, the True_Concrete threshold values are to examine this feature and then compared against True_Concrete models using the corpus dataset for the same and across data sizes.

The Effect of using Referring Expressions as Input text

In order to investigate the effect of using referring expressions versus the corpus dataset with all other training parameters as constant and independent from adding the visual embeddings, All_bstract threshold values are used for this analysis, where approximately all the words have only contextual embeddings. As a baseline, these results are compared against the ELECTRA-Xsmall model as the word embeddings are purely learned from text data. In addition, this sub-question is investigated across all three data sizes. The models examined here are: ELECTRA-Xsmall, mix_AoA_2, mix_AoA_3 and mix_concreteness_5. The results are shown in Tables 4.3, 4.4, 4.5 and 4.6.

Across The Same Data Size

According to SST-2 results, there are no significant differences in terms of accuracy between the corpus and referring expressing dataset for Size1 of the data. However, for Size2 and Size3, the models trained on referring expressions outperform the ones trained on the corpus dataset. The accuracy difference increases with data size; the difference ranges between 0 to 0.2 for Size2, with the mix_AoA_2 model scoring 0

difference. With respect to Size3, the difference in accuracy is stable at 0.3. The MNLI results are similar to the SST-2 results, except that for all corpus pre-trained models, the accuracy is below 0.5, and for the referring expressions pre-trained models, the accuracy is only above 0.5 for the Size3 models. Consequently, the results for Size1 demonstrate no difference in terms of accuracy. Nevertheless, there is a difference that ranges between 0 to 0.12 for Size2 and 0.26 to 0.27 for Size3. The RTE results indicate that there is no difference in accuracy between the corpus and the referring expression for Size1 and Size2; however, there is a small difference ranging from 0.03 to 0.06 for Size3. Surprisingly, the WNLI results indicate that the model accuracy for Size1 and Size3 has not changed, with model mix_AoA_2 differing by -0.17 for the corpus model. Despite this, Size2 follows the rule of the other tasks with a difference of 0.13. Similar to the MNLI results, the accuracy of the corpus-trained models is below 0.5 for both Size1 and Size2 in the data. All models report results that approximate the ones reported by the ELECTRA-Xsmall model (with equivalent data size), which applies to all the GLUE tasks.

Across Different Data Sizes

For the SST-2, MNLI, and RTE tasks, models that pre-trained on referring expression of Size2 outperform models trained on Size3 using corpus data, with a difference in accuracy ranging from 0.0 to 0.2 for the MNLI, from 0.0 to 0.2 for the SST-2 and 0.0 to 0.12 for the RTE. However, the MNLI accuracy is below 0.5 for all data sizes, which is unreliable. For The WNLI tasks, Size1 outperforms Size2, as the accuracy of the Size2 corpus-trained model is decreased.

Discussion

Based on the previous results, using referring expressions dataset outperformed using corpus dataset as the input text, for the same data size. This effect is more clear in the SST-2 task while it exists in RTE and MNLI but is only distinguished when using Size3. It is noted that all the task results have no difference in Size1. The referring expressions did not affect the WNLI. The results also show that using the referring expression as input data outperforms models pre-trained using larger corpus data, which implies that using such input data can compensate for using a smaller data size. However, this only works for data size larger than Size1, which also implies that there is a minimum size of data for this rule to be correct. The results from SST-2, WNLI, and RTE support this conclusion, while it is also true for the MNLI task. Nonetheless, the MNLI results are of a very low accuracy (below 0.5).

Limitations

The limitation here is the small model size (batch_size and max_sequence_length), as the length of the sentences in referring expressions data is small (avg of 3.4 for RefCOCO(Split=Train)), and the rest is padded with zeros, but a small max_sequence_length is used. On the contrary, with a large model, the max_sequence_length is 128, which means a lot more tokens will be zeros, which might affect the results for larger models.

The Effect of Using Coupled Image-Text Referring Expressions

In order to answer the second sub-question, the same method is followed to answer the previous one, which states comparing the results from using a corpus and referring expressions datasets as non-coupled and coupled data, respectively. The corpus text data and the images from RefCOCO and VisDial are considered non-coupled

Image-Text data, as they are extracted from different sources, and the text within the corpus does not —directly— refer to objects within the physical world. For analysis, models with visual embeddings added to their pre-training are compared. Thus, the True_Concrete threshold values are used, and the results from an ELECTRA-Xsmall model training are added as a baseline to compare the True_Concrete models. The models used in this analysis are: mix_AoA_4, mi_ Ao_5, mix_AoA_6, mix_AoA_7, mix_concreteness_3, mix_concreteness_4 and ELECTRA-Xsmall. The results are shown in Tables: 4.7, 4.8, 4.9, and 4.10.

Across the Same Data Size

For the SST-2 task, there is no difference in accuracy between the two Size1 model types. In contrast, the change for Size2 and Size3 increases from 0.0 to 0.5 for Size2 and from 0.0 to 0.3 for Size3, with models trained using coupled data as the upper hand. The same rule applies to the MNLI task, with an accuracy variance between 0.02 and 0.19. The RTE tasks report comparable accuracy for the two model types for Size1 and Size2, with a small difference ranging from 0.0 to 0.06 for Size3. In contrast to the previous tasks, the WNLI tasks report an accuracy change of 0.13 for both Size1 and Size2. In contrast, Size3 exhibits the opposite effect, with an accuracy difference of -0.13, as the non-coupled models (corpus-based models) reported higher accuracy for this size compared to stable and lower accuracy for the coupled referring expressions models.

Across Different Data Sizes

For both the SST-2 and MNLI, models trained on coupled data of Size2 outperform models trained on non-coupled data of Size3, but it does not apply for Size1 with

Size2. However, the WNLI task shows that models trained on coupled data of Size1 outperform models trained on non-coupled data of Size3, but this rule does not apply for Size2 with Size3. The results from the RTE task do not follow this rule, as there is a non-significant change in accuracy for the models with different data sources (coupled vs. non-coupled data) and different data sizes.

Discussion

For the same data size, the result is no change for Size1, indicating that the small size data is indifferent to using the coupled data vs. non-coupled data. For Size2, the RTE task is also indifferent, but other tasks MNLI and SST-2, have a slight difference with models pre-trained using coupled data. This rule applies to Size3 and for SST-2, MNLI and RTE tasks. However, the most significant difference in accuracy is observed between the ELECTRA-Xsmall model pre-trained on corpus data, which implies that the change I notice is not due to the coupled data effect, as ELECTRA-Xsmall pre-training does not involve any visual embeddings.

Data_Size	Model_Name	Corpus	Referring Expressions
Size1	ELECTRA-Xsmall	0.509174	0.509174
	mix_AoA_4	0.509174	0.509174
	mix_AoA_5	0.509174	0.509174
	mix_AoA_6	0.509174	0.509174
	mix_AoA_7	0.509174	0.509174
	mix_concreteness_3	0.509174	0.509174
	mix_concreteness_4	0.509174	0.509174
Size2	ELECTRA-Xsmall	0.509174	0.707569
	mix_AoA_4	0.509174	0.526376
	mix_AoA_5	0.509174	0.530963
	mix_AoA_6	0.509174	0.552752
	mix_AoA_7	0.509174	0.516055
	mix_concreteness_3	0.509174	0.509174
	mix_concreteness_4	0.509174	0.529817
Size3	ELECTRA-Xsmall	0.509174	0.813073
	mix_AoA_4	0.509174	0.813073
	mix_AoA_5	0.509174	0.808486
	mix_AoA_6	0.509174	0.811927
	mix_AoA_7	0.509174	0.815367
	mix_concreteness_3	0.509174	0.807339
	mix_concreteness_4	0.509174	0.801606

Table 4.7: The SST-2 results for using the coupled vs non-coupled data in pre-training using True_Concrete threshold models and the ELECTRA-Xsmall model which have the same settings except for freezing the word embeddings layer feature.

Data_Size	Model_Name	Corpus	Referring Expressions
Size1	ELECTRA-Xsmall	0.318186	0.318186
	mix_AoA_4	0.318186	0.318186
	mix_AoA_5	0.318186	0.318186
	mix_AoA_6	0.318186	0.318186
	mix_AoA_7	0.354457	0.318186
	mix_concreteness_3	0.318186	0.318288
	mix_concreteness_4	0.318186	0.318186
Size2	ELECTRA-Xsmall	0.318186	0.439531
	mix_AoA_4	0.318186	0.490372
	mix_AoA_5	0.318186	0.504534
	mix_AoA_6	0.318186	0.339786
	mix_AoA_7	0.318186	0.361488
	mix_concreteness_3	0.318186	0.354254
	mix_concreteness_4	0.318186	0.363118
Size3	ELECTRA-Xsmall	0.354457	0.585430
	mix_AoA_4	0.318186	0.584921
	mix_AoA_5	0.318186	0.577993
	mix_AoA_6	0.318186	0.588996
	mix_AoA_7	0.354457	0.582680
	mix_concreteness_3	0.318186	0.562303
	mix_concreteness_4	0.318186	0.576261

Table 4.8: The MNLI results for using the coupled vs non-coupled data in pre-training using True_Concrete threshold models and the ELECTRA-Xsmall model which have the same settings except for freezing the word embeddings layer feature.

Data_Size	Model_Name	Corpus	Referring Expressions
Size1	**ELECTRA-Xsmall**	0.527076	0.527076
	mix_AoA_4	0.527076	0.527076
	mix_AoA_5	0.527076	0.527076
	mix_AoA_6	0.527076	0.527076
	mix_AoA_7	0.527076	0.527076
	mix_concreteness_3	0.527076	0.527076
	mix_concreteness_4	0.527076	0.527076
Size2	**ELECTRA-Xsmall**	0.527076	0.527076
	mix_AoA_4	0.527076	0.530686
	mix_AoA_5	0.527076	0.527076
	mix_AoA_6	0.527076	0.527076
	mix_AoA_7	0.527076	0.527076
	mix_concreteness_3	0.472924	0.527076
	mix_concreteness_4	0.527076	0.527076
Size3	**ELECTRA-Xsmall**	0.523466	**0.545126**
	mix_AoA_4	0.472924	0.527076
	mix_AoA_5	0.472924	0.490975
	mix_AoA_6	0.527076	0.537906
	mix_AoA_7	0.527076	0.530686
	mix_concreteness_3	0.527076	0.509025
	mix_concreteness_4	0.527076	**0.545126**

Table 4.9: The RTE results for using the coupled vs non-coupled data in pre-training using True_Concrete threshold models and the ELECTRA-Xsmall model which have the same settings except for freezing the word embeddings layer feature.

Data_Size	Model_Name	Corpus	Referring Expressions
Size1	**ELECTRA-Xsmall**	0.43662	0.56338
	mix_AoA_4	0.43662	0.56338
	mix_AoA_5	0.43662	0.56338
	mix_AoA_6	0.43662	0.56338
	mix_AoA_7	0.43662	0.56338
	mix_concreteness_3	0.56338	0.56338
	mix_concreteness_4	0.43662	0.56338
Size2	**ELECTRA-Xsmall**	0.43662	0.56338
	mix_AoA_4	0.43662	0.56338
	mix_AoA_5	0.43662	0.56338
	mix_AoA_6	0.43662	0.56338
	mix_AoA_7	0.43662	0.56338
	mix_concreteness_3	0.43662	0.56338
	mix_concreteness_4	0.43662	0.56338
Size3	**ELECTRA-Xsmall**	0.56338	0.563380
	mix_AoA_4	0.56338	**0.436620**
	mix_AoA_5	0.56338	0.563380
	mix_AoA_6	0.56338	0.507042
	mix_AoA_7	0.56338	**0.436620**
	mix_concreteness_3	0.56338	0.563380
	mix_concreteness_4	0.56338	0.563380

Table 4.10: The WNLI results for using the coupled vs non-coupled data in pre-training using True_Concrete threshold models and the ELECTRA-Xsmall model which have the same settings except for freezing the word embeddings layer feature.

4.3.3 The Effect of adding Concreteness Distinction Knowledge

To answer the third question, which studies the effect of concreteness knowledge, the performance of the various threshold models, from All_Concrete, True_Concrete, and All_Abstract categories, are compared against one another, and with the ELECTRA-Xsmall model as a baseline. The results are examined in terms of data size and source, while considering that the OpenWebText corpus data is non-coupled with the visual embeddings. Therefore, conclusions are based only on the referring expressions data but report both. In this experiment, the analysis of the AoA and Concreteness Score datasets is separated, which should have the same pattern because they correlate together, as shown in Figure 3.5. The results are examined and summarized for this question in Tables: 4.11, 4.12, 4.13 and 4.14.

Results

According to the findings, there are a variety of variations ranging from a distinct difference to complete stability for all threshold categories across all data sizes. For corpus data, the threshold category made either no difference or random fluctuations in model performance across all data sizes. However, according to the referring expressions input data, Size1 and Size3 do not vary across threshold categories. In contrast, for Size2, the ELECTRA-Xsmall outperformed all other models, while the True_Concrete model mix_AoA_8's inaccuracy is only -0.2, which is the second highest across all thresholds categories. These outcomes are displayed in Table 4.11. The Concreteness Score thresholds follow the same pattern for Size1 and Size3, but with Size2 ELECTRA-Xsmall scored the highest, which is equivalent to the model

at mix_concreteness_5 threshold, which is an All_Abstract threshold, as depicted in Table 4.15.

For the MNLI task, results remained relatively unchanged. As with the SST-2 case, Size1 and Size3 have nearly identical results across all threshold values. However, Size2 performs better in mix_AoA_4 and mix_AoA_5, which are both from the True_Concrete category, with an accuracy difference of 0.07 compared to the ELECTRA-Xsmall, as shown in Table 4.12. The concreteness score thresholds reveal that the mix_concreteness_5 threshold has the highest performance of 0.43, which is equivalent to ELECTRA-Xsmall. Those are shown in Table 4.16,

As shown in Tables 4.13 and 4.17, The RTE also displays fluctuating results from all three threshold categories for referring expressions, Size1 and Size2 are both indifferent, but Size3 has slight variance with the highest score at the mix_AoA_11, which is an All_Concrete threshold, with a difference in the accuracy of 0.1 compared to ELECTRA-Xsmall.

The results of the WNLI task for the models trained on referring expressions models randomly fluctuate for Size3 but are stable for Size1 and Size2, which is also the pattern reported for the concreteness score thresholds. Those results are shown in Tables 4.14 and 4.18.

Discussion

According to the results above, models pre-trained on corpus datasets are indifferent to all threshold values, and their results are approximately identical to those of the ELECTRA-Xsmall model. The models trained on referring expressions produced different results, and their threshold values varied, but the upper bound is consistently scored by ELECTRA-Xsmall, which is the case for Size1 and Size3 of the data.

However, for Size2, the True_Concrete threshold performs slightly higher than the ELECTRA-Xsmall using the MNLI task and only according to the AoA thresholds. The SST-2 tasks show that the All_Abstract or ELECTRA-Xsmall outperforms the others. The other RTE and WNLI have fluctuating results, so they are not valid for this analysis. Except for the cases mentioned, the other model's performance does not change with threshold values, which means it does not change with concreteness distinction knowledge (adding concreteness distinction knowledge does not enhance the performance of the abstract ELECTRA-Xsmall model).

Limitations

The occasional inconsistency between the AoA thresholds results and corresponding Concreteness score results may be due to the intersection between both datasets and RefCOCO and VisDial datasets and OpenWebText, as depicted in Tables 3.3 and 3.2.

Data_Size	Concreteness Threshold	Model_Name	Corpus	Referring Expressions
Size1	All_Abstract	ELECTRA-Xsmall	0.509174	**0.509174**
		mix_AoA_2	0.509174	0.509174
		mix_AoA_3	0.509174	0.509174
	True_Concrete	mix_AoA_4	0.509174	0.509174
		mix_AoA_5	0.509174	0.509174
		mix_AoA_6	0.509174	0.509174
		mix_AoA_7	0.509174	0.509174
	All_Concrete	mix_AoA_8	0.509174	0.509174
		mix_AoA_9	0.509174	0.509174
		mix_AoA_10	0.509174	0.509174
		mix_AoA_11	0.509174	0.509174

(a)

Data_Size	Concreteness Threshold	Model_Name	Corpus	Referring Expressions
Size2	All_Abstract	ELECTRA-Xsmall	0.509174	**0.707569**
		mix_AoA_2	0.509174	0.509174
		mix_AoA_3	0.509174	**0.600917**
	True_Concrete	mix_AoA_4	0.509174	0.526376
		mix_AoA_5	0.509174	0.530963
		mix_AoA_6	0.509174	0.552752
		mix_AoA_7	0.509174	0.516055
	All_Concrete	mix_AoA_8	0.509174	**0.686927**
		mix_AoA_9	0.509174	0.508028
		mix_AoA_10	0.509174	0.509174
		mix_AoA_11	0.509174	0.509174

(b)

Data_Size	Concreteness Threshold	Model_Name	Corpus	Referring Expressions
Size3	All_Abstract	ELECTRA-Xsmall	0.509174	**0.813073**
		mix_AoA_2	0.509174	0.805046
		mix_AoA_3	0.509174	0.814220
	True_Concrete	mix_AoA_4	0.509174	0.813073
		mix_AoA_5	0.509174	0.808486
		mix_AoA_6	0.509174	0.811927
		mix_AoA_7	0.509174	0.815367
	All_Concrete	mix_AoA_8	0.509174	0.807339
		mix_AoA_9	0.509174	0.810780
		mix_AoA_10	0.509174	0.807339
		mix_AoA_11	0.509174	0.813073

(c)

Table 4.11: The SST-2 results for using all three different concreteness threshold categories from the AoA dataset and ELECTRA-Xsmall across the three data sizes: Size1, Size2, and Size3.

Data_Size	Concreteness Threshold	Model_Name	Corpus	Referring Expressions
Size1	All_Abstract	ELECTRA-Xsmall	0.318186	**0.318186**
		mix_AoA_2	0.318186	0.318186
		mix_AoA_3	0.318186	0.318186
	True_Concrete	mix_AoA_4	0.318186	0.318186
		mix_AoA_5	0.318186	0.318186
		mix_AoA_6	0.318186	0.318186
		mix_AoA_7	0.354457	0.318186
	All_Concrete	mix_AoA_8	0.318186	0.318186
		mix_AoA_9	0.318186	0.318696
		mix_AoA_10	0.318186	0.354457
		mix_AoA_11	0.354457	0.318186

(a)

Data_Size	Concreteness Threshold	Model_Name	Corpus	Referring Expressions
Size2	All_Abstract	ELECTRA-Xsmall	0.318186	**0.439531**
		mix_AoA_2	0.318186	0.318186
		mix_AoA_3	0.318186	0.353439
	True_Concrete	mix_AoA_4	0.318186	**0.490372**
		mix_AoA_5	0.318186	**0.504534**
		mix_AoA_6	0.318186	0.339786
		mix_AoA_7	0.318186	0.361488
	All_Concrete	mix_AoA_8	0.318186	0.328069
		mix_AoA_9	0.318186	0.318186
		mix_AoA_10	0.318186	0.318186
		mix_AoA_11	0.318186	0.318186

(b)

Data_Size	Concreteness Threshold	Model_Name	Corpus	Referring Expressions
Size3	All_Abstract	ELECTRA-Xsmall	0.354457	**0.585430**
		mix_AoA_2	0.318186	0.586449
	True_Concrete	mix_AoA_3	0.318186	0.571065
		mix_AoA_4	0.318186	0.584921
		mix_AoA_5	0.318186	0.577993
		mix_AoA_6	0.318186	0.588996
		mix_AoA_7	0.354457	0.582680
	All_Concrete	mix_AoA_8	0.318186	0.552522
		mix_AoA_9	0.318186	0.575853
		mix_AoA_10	0.318186	0.549771
		mix_AoA_11	0.318186	0.577076

(c)

Table 4.12: The MNLI results for using all three different concreteness threshold categories from the AoA dataset and ELECTRA-Xsmall across the three data sizes: Size1, Size2, and Size3.

Data_Size	Concreteness Threshold	Model_Name	Corpus	Referring Expressions
Size1	All_Abstract	**ELECTRA-Xsmall**	0.527076	**0.527076**
		mix_AoA_2	0.527076	0.527076
		mix_AoA_3	0.527076	0.527076
		mix_AoA_4	0.527076	0.527076
	True_Concrete	**mix_AoA_5**	0.527076	0.527076
		mix_AoA_6	0.527076	0.527076
		mix_AoA_7	0.527076	0.527076
		mix_AoA_8	0.527076	0.527076
	All_Concrete	**mix_AoA_9**	0.527076	0.527076
		mix_AoA_10	0.527076	0.527076
		mix_AoA_11	0.527076	0.472924

(a)

Data_Size	Concreteness Threshold	Model_Name	Corpus	Referring Expressions
Size2	All_Abstract	**ELECTRA-Xsmall**	0.527076	**0.527076**
		mix_AoA_2	0.527076	0.527076
		mix_AoA_3	0.527076	0.530686
		mix_AoA_4	0.527076	0.530686
	True_Concrete	**mix_AoA_5**	0.527076	0.527076
		mix_AoA_6	0.527076	0.527076
		mix_AoA_7	0.527076	0.527076
	All_Concrete	**mix_AoA_8**	0.527076	0.530686
		mix_AoA_9	0.527076	0.527076
		mix_AoA_10	0.527076	0.527076
		mix_AoA_11	0.527076	0.527076

(b)

Data_Size	Concreteness Threshold	Model_Name	Corpus	Referring Expressions
Size3	All_Abstract	**ELECTRA-Xsmall**	0.523466	**0.545126**
		mix_AoA_2	0.527076	0.530686
		mix_AoA_3	0.472924	0.527076
		mix_AoA_4	0.472924	0.527076
	True_Concrete	**mix_AoA_5**	0.472924	0.490975
		mix_AoA_6	0.527076	0.537906
		mix_AoA_7	0.527076	0.530686
		mix_AoA_8	0.472924	0.530686
	All_Concrete	**mix_AoA_9**	0.472924	0.534296
		mix_AoA_10	0.472924	0.505415
		mix_AoA_11	0.527076	0.555957

(c)

Table 4.13: The RTE results for using all three different concreteness threshold categories from the AoA dataset and ELECTRA-Xsmall across the three data sizes: Size1, Size2, and Size3.

Data_Size	Concreteness Threshold	Model_Name	Corpus	Referring Expressions
Size1	All_Abstract	**ELECTRA-Xsmall**	0.43662	**0.56338**
		mix_AoA_2	0.56338	0.56338
		mix_AoA_3	0.56338	0.56338
	True_Concrete	**mix_AoA_4**	0.43662	0.56338
		mix_AoA_5	0.43662	0.56338
		mix_AoA_6	0.43662	0.56338
		mix_AoA_7	0.43662	0.56338
	All_Concrete	**mix_AoA_8**	0.43662	0.56338
		mix_AoA_9	0.43662	0.56338
		mix_AoA_10	0.56338	0.43662
		mix_AoA_11	0.43662	0.56338

(a)

Data_Size	Concreteness Threshold	Model_Name	Corpus	Referring Expressions
Size2	All_Abstract	**ELECTRA-Xsmall**	0.43662	**0.56338**
		mix_AoA_2	0.43662	0.56338
		mix_AoA_3	0.43662	0.56338
	True_Concrete	**mix_AoA_4**	0.43662	0.56338
		mix_AoA_5	0.43662	0.56338
		mix_AoA_6	0.43662	0.56338
		mix_AoA_7	0.43662	0.56338
	All_Concrete	**mix_AoA_8**	0.43662	0.56338
		mix_AoA_9	0.43662	0.56338
		mix_AoA_10	0.43662	0.56338
		mix_AoA_11	0.43662	0.56338

(b)

Data_Size	Concreteness Threshold	Model_Name	Corpus	Referring Expressions
Size3	All_Abstract	**ELECTRA-Xsmall**	0.56338	**0.563380**
		mix_AoA_2	0.56338	0.394366
		mix_AoA_3	0.56338	0.563380
	True_Concrete	**mix_AoA_4**	0.56338	0.436620
		mix_AoA_5	0.56338	0.563380
		mix_AoA_6	0.56338	0.507042
		mix_AoA_7	0.56338	0.436620
	All_Concrete	**mix_AoA_8**	0.56338	0.436620
		mix_AoA_9	0.56338	0.422535
		mix_AoA_10	0.56338	0.563380
		mix_AoA_11	0.56338	0.436620

(c)

Table 4.14: The WNLI results for using all three different concreteness threshold categories from the AoA dataset and ELECTRA-Xsmall across the three data sizes: Size1, Size2, and Size3.

Data_Size	Concreteness Threshold	Model_Name	Corpus	Referring Expressions
Size1	All_Concrete	**mix_concreteness_1**	0.509174	0.509174
		mix_concreteness_2	0.509174	0.509174
	True_Concrete	**mix_concreteness_3**	0.509174	0.509174
		mix_concreteness_4	0.509174	0.509174
	All_Abstract	**mix_concreteness_5**	0.509174	0.497706
		ELECTRA-Xsmall	0.509174	**0.509174**
Size2	All_Concrete	**mix_concreteness_1**	0.509174	0.522936
		mix_concreteness_2	0.509174	0.587156
	True_Concrete	**mix_concreteness_3**	0.509174	0.509174
		mix_concreteness_4	0.509174	0.529817
	All_Abstract	**mix_concreteness_5**	0.509174	0.707569
		ELECTRA-Xsmall	0.509174	**0.707569**
Size3	All_Concrete	**mix_concreteness_1**	0.509174	0.817661
		mix_concreteness_2	0.509174	0.815367
	True_Concrete	**mix_concreteness_3**	0.509174	0.807339
		mix_concreteness_4	0.509174	0.801606
	All_Abstract	**mix_concreteness_5**	0.509174	0.816514
		ELECTRA-Xsmall	0.509174	**0.813073**

Table 4.15: The SST-2 results for using all three different concreteness threshold categories from the Concreteness Score dataset and ELECTRA-Xsmall across the three data size: Size1, Size2, and Size3.

Data_Size	Concreteness Threshold	Model_Name	Corpus	Referring Expressions
Size1	All_Concrete	**mix_concreteness_1**	0.318186	0.372084
		mix_concreteness_2	0.318186	0.318186
	True_Concrete	**mix_concreteness_3**	0.318186	0.318288
		mix_concreteness_4	0.318186	0.318186
	All_Abstract	**mix_concreteness_5**	0.318186	0.318186
		ELECTRA-Xsmall	0.318186	**0.318186**
Size2	All_Concrete	**mix_concreteness_1**	0.318186	0.333164
		mix_concreteness_2	0.318186	0.338971
	True_Concrete	**mix_concreteness_3**	0.318186	0.354254
		mix_concreteness_4	0.318186	0.363118
	All_Abstract	**mix_concreteness_5**	0.318186	0.439531
		ELECTRA-Xsmall	0.318186	**0.439531**
Size3	All_Concrete	**mix_concreteness_1**	0.318186	0.582170
		mix_concreteness_2	0.318186	0.582680
	True_Concrete	**mix_concreteness_3**	0.318186	0.562303
		mix_concreteness_4	0.318186	0.576261
	All_Abstract	**mix_concreteness_5**	0.354457	0.573917
		ELECTRA-Xsmall	0.354457	**0.585430**

Table 4.16: The MNLI results for using all three different concreteness threshold categories from the Concreteness Score dataset and ELECTRA-Xsmall across the three data size: Size1, Size2, and Size3.

Data_Size	Concreteness Threshold	Model_Name	Corpus	Referring Expressions
Size1	All_Concrete	**mix_concreteness_1**	0.527076	0.527076
		mix_concreteness_2	0.527076	0.527076
	True_Concrete	**mix_concreteness_3**	0.527076	0.527076
		mix_concreteness_4	0.527076	0.527076
	All_Abstract	**mix_concreteness_5**	0.527076	0.527076
		ELECTRA-Xsmall	0.527076	**0.527076**
Size2	All_Concrete	**mix_concreteness_1**	0.527076	0.527076
		mix_concreteness_2	0.527076	0.527076
	True_Concrete	**mix_concreteness_3**	0.472924	0.527076
		mix_concreteness_4	0.527076	0.527076
	All_Abstract	**mix_concreteness_5**	0.527076	0.527076
		ELECTRA-Xsmall	0.527076	**0.527076**
Size3	All_Concrete	**mix_concreteness_1**	0.472924	0.530686
		mix_concreteness_2	0.472924	0.530686
	True_Concrete	**mix_concreteness_3**	0.527076	0.509025
		mix_concreteness_4	0.527076	0.545126
	All_Abstract	**mix_concreteness_5**	0.472924	0.498195
		ELECTRA-Xsmall	0.523466	**0.545126**

Table 4.17: The RTE results for using all three different concreteness threshold categories from the Concreteness Score dataset and ELECTRA-Xsmall across the three data size: Size1, Size2, and Size3.

Data_Size	Concreteness Threshold	Model_Name	Corpus	Referring Expressions
Size1	All_Concrete	**mix_concreteness_1**	0.43662	0.56338
		mix_concreteness_2	0.43662	0.56338
	True_Concrete	**mix_concreteness_3**	0.56338	0.56338
		mix_concreteness_4	0.43662	0.56338
	All_Abstract	**mix_concreteness_5**	0.43662	0.56338
		ELECTRA-Xsmall	0.43662	**0.56338**
Size2	All_Concrete	**mix_concreteness_1**	0.43662	0.56338
		mix_concreteness_2	0.43662	0.56338
	True_Concrete	**mix_concreteness_3**	0.43662	0.56338
		mix_concreteness_4	0.43662	0.56338
	All_Abstract	**mix_concreteness_5**	0.43662	0.56338
		ELECTRA-Xsmall	0.43662	**0.56338**
Size3	All_Concrete	**mix_concreteness_1**	0.56338	0.563380
		mix_concreteness_2	0.56338	0.436620
	True_Concrete	**mix_concreteness_3**	0.56338	0.563380
		mix_concreteness_4	0.56338	0.563380
	All_Abstract	**mix_concreteness_5**	0.56338	0.577465
		ELECTRA-Xsmall	0.56338	**0.563380**

Table 4.18: The WNLI results for using all three different concreteness threshold categories from the Concreteness Score dataset and ELECTRA-Xsmall across the three data size: Size1, Size2, and Size3.

4.3.4 The Effect of Data Size

Although this question does not have a direct implication in the hypothesis, the

effect of data size on the current experiments is further investigated, as it is one of the limitations encountered due to the small data size used. For this analysis, the results shown in Tables 4.11, 4.12, 4.13, 4.14, 4.15, 4.16, 4.17, and 4.18 are investigated.

Results

For the STS-2, the size does not affect models pre-trained on corpus dataset. However, for the ELECTRA-Xsmall model pre-trained on the referring expressions, the STS-2 and MNLI tasks witness an increasing change in accuracy from 0.5, 0.7, 0.8 and 0.3, 0.4, 0.58 for Size1, Size2 and Size3, respectively. However, for the RTE and WNLI tasks, there is no change in the performance of ELECTRA-Xsmall model with changing the data size.

Discussion

For the models trained on referring expression, it is evident that the data size has a significant impact on the results only for the SST-2 and MNLI tasks, whereas the RTE and WNLI tasks experience no/insignificant change. This pattern is reported by the. In contrast, in the models trained using text corpus data, there is no effect for the size on the performance and all the tasks used.

4.4 Results Discussion

Summing up the results from the previous analysis, the addition of visual embeddings did not improve the performance of the corpus-trained models, and this holds true for all data sizes. The same rule applies to models that have been pre-trained on referring expression, except for Size2 reported by MNLI tasks. The models that have been pre-trained with Size1 are indifferent to the use of referring expressions or

corpus data. With Size2 and Size3, however, performance significantly improves with the SST-2 task, whereas MNLI and RTE have only a slight performance increase that develops with the data size. In addition, models pre-trained with Size2 of referring expressions data outperform models pre-trained with Size3 corpus data. This result is due to the stable performance of models using corpus text as pre-training input data, while models pre-trained with referring expression increase in performance with increased data size. From the previous analysis, it is difficult to conclude that using coupled Image-Text data enhances the model performance as adding the visual embeddings does not affect the performance, which makes me lean towards the the improvement from using the coupled Image-Text referring expressions over the non-coupled Image-Text corpus is mainly due to using the referring expressions, and the couple/non-coupled data has no effect over it. I support this decision by comparing my results to the ELECTRA- Xsmall models (trained using the same data source and size), as this model does not have any visual word embeddings. The results imply that the models pre-trained on both the corpus and referring expressions text data are not affected by adding the concreteness knowledge, as varying the concreteness threshold from the All_Abstract conditions to the All_Concrete conditions does not yield a distinct variation in performance. Moreover, comparing the results to the ELECTRA-Xsmall (with the same parameters) affirms the previous results, except for the True_Concrete models pre-trained on Size2 of referring expressions data that outperform the other models (also pre-trained on Size2 of referring expressions text data), reported by the MNLI metric.

Finally, the current analysis results are compared with those reported in the previous literature (i.e., ELECTRA-small). Table 4.2 compares the results of the

ELECTRA-Xsmall trained on referring expressions of Size3. For the SST-2, MNLI, RTE and WNLI the results are: 0.87, 0.79, 0.64, 0.53 and 0.81, 0.59, 0.55, 0.56 for ELECTRA-small (trained on OpenWebText of size 40 GB) and ELECTRA-Xsmall (trained on Size3 of 16.9 MB referring expressions) respectively. The results are comparable, with ELECTRA-small outperforming ELECTRA-Xsmall, but the data size each model is trained on is incomparable, indicating that this approach has the potential to be further investigated in future work, especially when the data size is critical to the application and a smaller model size is used to incorporate the data limits.

4.5 Limitations

After exploring the results and evaluating the current work, the limitations are summarized, contributing to interpretations of the findings and highlighting the limitations.

4.5.1 Data Limitations

The used data probably have a significant number of limitations. The first one is the small data size, as all LMs (such as BERT or ELECTRA) use massive data, which enables the model to report high performance when fine-tuned. In addition, it is possible to fine-tune small data sizes and yet have good performance, and the larger the data size, the larger the performance. Due to the small data size, the model size is reduced (another limitation). The second limitation is using VisDial datasets' answers, which is noisy for referring expressions as the sentences (answers) are in the context of the dialogue, and not all of the answers are descriptive of the image, but some of them could be just follow-up to a previous question/answer. Another data limitation is the combination of both RefCOCO and VisDial datasets, as those are

collected from different settings and for different tasks (explained in Section 2.5 and Section 2.6), and the difference between the images from both datasets, as images from the RefCOCO dataset are the sub-image (object within the original image with relative position information). In contrast, the VisDial images are the whole image with more than one object, so it does not have position information (position vector is zeros). A fourth limitation is the intersection of the words between words within the concreteness knowledge dataset and words with WAC visual embeddings (derived from the coupled Image-Text datasets). For instance, the interaction between words in the AoA dataset and the words with WAC embeddings for data sizes: Size1, Size2, and Size3 are: 30%, 32%, and 39%, respectively. Similarly, the intersection with the Concreteness Threshold dataset is 40%, 39%, and 50%, respectively, which means that less than half of words within the concreteness knowledge datasets have WAC visual embeddings, and this percentage decrease with choosing different concreteness thresholds that lean toward being True_Concrete thresholds. These results are summarized in Table3.2 and Table 3.3 (for Concreteness Score and AoA datasets, respectively).

4.5.2 ELECTRA-Xsmall limitations

Using a smaller version of the ELECTRA-small model is essential to work with the small-size text data limitation, but it reduces the model performance, as shown in Table 4.2, using the OpenWebText data with smaller batch_size and max_seq_length values. Another limitation regarding pre-training the model is that swapping the embeddings while freezing creates a gap between the concrete and abstract word embeddings as both are learned from entirely different sources (text and images).

4.5.3 WAC Limitations

Although the WAC model has a number of limitations (it assumes all words have only visual meaning and sentence structure is ignored), the limitations affecting the current results are only considered. One of those limitations is that the WAC model creates embeddings for words that occurred more than a certain number of times within the images-text dataset, which is not a suitable indication that this word is concrete (and should have visual embeddings). These words can be "the", "at", "is", which are very misleading and, therefore, give the All_Concrete conditions more false visual embeddings (visual meanings). Another limitation of the same effect is that WAC assumes all words are concrete, so another filtration method needs to be used to balance this effect. However, this limitation does not affect the True_Concrete and the All_Abstract thresholds models.

4.5.4 Mixing The Embeddings Limitations

Due to the gaps within the intersection between the concreteness distinction datasets used and words with WAC embeddings (extracted from textual pre-training data), there will be special cases for words that are determined to be concrete (by the threshold) but do not have WAC visual embeddings (either do not meet the WAC conditions to create classifier or do not exist in the RfCOCO/VisDial dataset), so these words will have an embedding as a vector of zeros. This limitation is exacerbated by freezing the weights (due to ELECTRA pre-training), as it prevents changing the value of the embedding but instead changes the Generator and Discriminator parameters.

4.5.5 GLUE limitations

Although the GLUE benchmark is widely used as a metric for Natural Language

understanding, it has limitations, some of which significantly impact the current work more than others. As previously mentioned, the small data size for specific tasks (such as RTE and WNLI) makes it unreliable for small models' fine-tuning. Another limitation is that tasks involving two sentences (such as QNLI and WNLI) are inapplicable to my work because the data consisted of only simple, separated sentences. Overall, GLUE only measures how models learn contextually, but learning the meanings requires additional tasks, such as robot-human interaction tasks, to evaluate performance.

4.6 Conclusion

The motivation behind this work is to advance LMS to the next step. Indeed, these models do an outstanding job learning some meanings for words from the text. However, there is a wide gap between the meaning that humans and the LMs learn for words, starting from the methodology humans learn their first language. Looking at the first years of language acquisition for children, the settings used are entirely different than how LMs work, yet if applied, it would solve the LMs issues, such as the wrong assumptions about the word's meaning —meaning is all abstract— and the need of large size of pre-training data. Therefore, this work aims to address these limitations for the LMs and further emulate similar settings in which children learn their first language to enrich the LMs, with the addition that the meaning of the word is both concrete for some words and abstract.

Using referring expressions helps the model approach higher accuracy for GLUE tasks. In addition, the difference is clearer with the higher data size. Using corpus-based data, however, is unaffected by data size (for the small data sizes I use), even when the model is pre-trained using the entire OpenWebText as shown in Table 4.2.

This implies that decreasing the model size prevents the model from learning the language (while all parameters, including the number of epochs, are the same), which is the effect observed when the batch_size and max_sequence_length parameters are both less than 128.

Additionally, the concreteness distinction knowledge did not improve the model's performance (except on the MNLI tasks in Size2). However, this is contested due to the small number of words with WAC visual embeddings (extracted from the text in RefCOCO and VisDial datasets) and words from AoA or Concreteness score datasets that intersect. As shown in Tables 3.2 and Table 3.3, the maximum intersection is less than fifty percent, indicating that more than 50 % of the words with visual WAC embeddings cannot be distinguished as concrete and that the model does not use these visual embeddings for these words. This finding implies that future research will require a higher intersection percentage and more text cleaning and preprocessing to close the gap (in terms of the number of words) between both datasets (i.e., stemming and lemmatization). In Kennington (2021), it was shown that adding visual embeddings to a model improved its accuracy. This is another reason why adding concreteness knowledge to a model would be beneficial. The difference is that they used ELECTRA-small (a more significant model than the one I use), the entire 40 GB of OpenWebText data, and visual embeddings for a much larger set of words than used in the current research, which is 27,152 words out of 30K words.

Finally, according to the current research hypothesis, adding visual knowledge of the world controlled by concreteness distinction knowledge and using the text of referring expressions to pre-train transformer-based language models will improve their performance with smaller text data sets (measured in Bytes). The results,

however, demonstrate that the methodology utilized in this thesis cannot prove the first two parts of the research hypothesis (adding visual knowledge and adding concreteness distinction knowledge will improve LMs). On the contrary, the results suggest that using referring expressions as the text input to the reduced-size ELECTRA-Xsmall model enhanced its performance when applied to pure language understanding benchmark tasks. The outcomes are encouraging for further research in this field, particularly to overcome the massive data size restriction imposed by transformer-based language models.

REFERENCES

Borghi, Anna M, Barca, Laura, Binkofski, Ferdinand, Castelfranchi, Cristiano, Pezzulo, Giovanni, & Tummolini, Luca. 2019. Words as social tools: Language, sociality and inner grounding in abstract concepts. *Phys. Life Rev.*, **29**(July), 120–153.

Brysbaert, M., Warriner, Amy Beth, & Kuperman, V. 2014. Concreteness ratings for 40 thousand generally known English word lemmas. *Behavior Research Methods*, **46**, 904–911.

Clark, Kevin, Luong, Minh-Thang, Le, Quoc V., & Manning, Christopher D. 2020. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. *ArXiv*, **abs/2003.10555**.

Das, Abhishek, Kottur, Satwik, Gupta, Khushi, Singh, Avi, Yadav, Deshraj, Moura, José M. F., Parikh, Devi, & Batra, Dhruv. 2017. Visual Dialog. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1080–1089.

Devlin, Jacob, Chang, Ming-Wei, Lee, Kenton, & Toutanova, Kristina. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Pages 4171–4186 of: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Tech-*

nologies, Volume 1 (Long and Short Papers). Minneapolis, Minnesota: Association for Computational Linguistics.

Gokaslan, Aaron, & Cohen, Vanya. 2019. *OpenWebText Corpus*.

Huang, Zhicheng, Zeng, Zhaoyang, Liu, Bei, Fu, Dongmei, & Fu, Jianlong. 2020. *Pixel-BERT: Aligning Image Pixels with Text by Deep Multi-Modal Transformers*.

Jiao, Xiaoqi, Yin, Yichun, Shang, Lifeng, Jiang, Xin, Chen, Xiao, Li, Linlin, Wang, Fang, & Liu, Qun. 2020. *TinyBERT: Distilling BERT for Natural Language Understanding*.

Kazemzadeh, Sahar, Ordonez, Vicente, Matten, M., & Berg, Tamara L. 2014. ReferItGame: Referring to Objects in Photographs of Natural Scenes. *In: EMNLP*.

Kennington, Casey. 2021. Enriching Language Models with Visually-grounded Word Vectors

and the Lancaster Sensorimotor Norms. *In: Proceedings of the 25th Conference on Computational Natural Language Learning*. Association for Computational Linguistics.

Kuperman, V., Stadthagen-González, H., & Brysbaert, M. 2012. Age-of-acquisition ratings for 30,000 English words. *Behavior Research Methods*, **44**, 978–990.

Levesque, Hector J., Davis, Ernest, & Morgenstern, Leora. 2012. The Winograd Schema Challenge. *Page 552–561 of: Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning*. KR'12. AAAI Press.

Liang, Zujie, Hu, Huang, Xu, Can, Tao, Chongyang, Geng, Xiubo, Chen, Yining, Liang, Fan, & Jiang, Daxin. 2021. *Maria: A Visual Experience Powered Conversational Agent.*

Lin, Tsung-Yi, Maire, Michael, Belongie, Serge J., Hays, James, Perona, Pietro, Ramanan, Deva, Dollár, Piotr, & Zitnick, C. Lawrence. 2014. Microsoft COCO: Common Objects in Context. *In: ECCV.*

Lu, Jiasen, Batra, Dhruv, Parikh, Devi, & Lee, Stefan. 2019. *ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks.*

McCune, Lorraine. 2008. *How Children Learn to Learn Language.* Oxford University Press.

McInnes, Leland, Healy, John, & Melville, James. 2018. *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction.*

Murahari, Vishvak, Batra, Dhruv, Parikh, Devi, & Das, Abhishek. 2020. *Large-scale Pretraining for Visual Dialog: A Simple State-of-the-Art Baseline.*

Peters, Matthew E., Neumann, Mark, Iyyer, Mohit, Gardner, Matt, Clark, Christopher, Lee, Kenton, & Zettlemoyer, Luke. 2018. Deep Contextualized Word Representations. *Pages 2227–2237 of: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers).* New Orleans, Louisiana: Association for Computational Linguistics.

Radford, Alec, & Narasimhan, Karthik. 2018. Improving Language Understanding by Generative Pre-Training.

Radford, Alec, Kim, Jong Wook, Hallacy, Chris, Ramesh, Aditya, Goh, Gabriel, Agarwal, Sandhini, Sastry, Girish, Askell, Amanda, Mishkin, Pamela, Clark, Jack, Krueger, Gretchen, & Sutskever, Ilya. 2021. Learning Transferable Visual Models From Natural Language Supervision. *In: ICML*.

Rajpurkar, Pranav, Zhang, Jian, Lopyrev, Konstantin, & Liang, Percy. 2016a. SQuAD: 100,000+ Questions for Machine Comprehension of Text. *In: EMNLP*.

Rajpurkar, Pranav, Zhang, Jian, Lopyrev, Konstantin, & Liang, Percy. 2016b. SQuAD: 100,000+ Questions for Machine Comprehension of Text. *Pages 2383–2392 of: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics.

Rogers, Anna, Kovaleva, Olga, & Rumshisky, Anna. 2021. A Primer in BERTology: What We Know About How BERT Works. *Transactions of the Association for Computational Linguistics*, **8**(01), 842–866.

Russakovsky, Olga, Deng, Jia, Su, Hao, Krause, Jonathan, Satheesh, Sanjeev, Ma, Sean, Huang, Zhiheng, Karpathy, Andrej, Khosla, Aditya, Bernstein, Michael, *et al.* 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision*, **115**(3), 211–252.

Schlangen, David, Zarriess, Sina, & Kennington, Casey. 2016. *Resolving References to Objects in Photographs using the Words-As-Classifiers Model*.

Sharma, Piyush, Ding, Nan, Goodman, Sebastian, & Soricut, Radu. 2018. Conceptual Captions: A Cleaned, Hypernymed, Image Alt-text Dataset For Automatic Image Captioning. *Pages 2556–2565 of: Proceedings of the 56th Annual Meeting of the*

Association for Computational Linguistics (Volume 1: Long Papers). Melbourne, Australia: Association for Computational Linguistics.

Smith, Linda, & Gasser, Michael. 2005. The Development of Embodied Cognition: Six Lessons from Babies. *Artificial Life*, 13–29.

Tan, Hao, & Bansal, Mohit. 2019. *LXMERT: Learning Cross-Modality Encoder Representations from Transformers*.

Vaswani, Ashish, Shazeer, Noam M., Parmar, Niki, Uszkoreit, Jakob, Jones, Llion, Gomez, Aidan N., Kaiser, Lukasz, & Polosukhin, Illia. 2017. Attention is All you Need. *ArXiv*, **abs/1706.03762**.

Wang, Alex, Singh, Amanpreet, Michael, Julian, Hill, Felix, Levy, Omer, & Bowman, Samuel R. 2019. *GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding*.

Williams, Adina, Nangia, Nikita, & Bowman, Samuel R. 2018. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. *In: NAACL*.

Wolf, Thomas, Debut, Lysandre, Sanh, Victor, Chaumond, Julien, Delangue, Clement, Moi, Anthony, Cistac, Pierric, Rault, Tim, Louf, Rémi, Funtowicz, Morgan, & Brew, Jamie. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *ArXiv*, **abs/1910.03771**.

Wu, Yonghui, Schuster, Mike, Chen, Z., Le, Quoc V., Norouzi, Mohammad, Macherey, Wolfgang, Krikun, Maxim, Cao, Yuan, Gao, Qin, Macherey, Klaus, Klingner, Jeff, Shah, Apurva, Johnson, Melvin, Liu, Xiaobing, Kaiser, Lukasz, Gouws, Stephan, Kato, Yoshikiyo, Kudo, Taku, Kazawa, Hideto, Stevens, Keith,

Kurian, George, Patil, Nishant, Wang, Wei, Young, Cliff, Smith, Jason R., Riesa, Jason, Rudnick, Alex, Vinyals, Oriol, Corrado, Gregory S., Hughes, Macduff, & Dean, Jeffrey. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *ArXiv*, **abs/1609.08144**.

Xu, Haiyang, Yan, Ming, Li, Chenliang, Bi, Bin, Huang, Songfang, Xiao, Wenming, & Huang, Fei. 2021. E2E-VLP: End-to-End Vision-Language Pre-training Enhanced by Visual Learning. *Pages 503–513 of: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Online: Association for Computational Linguistics.

Yin, Yichun, Chen, Cheng, Shang, Lifeng, Jiang, Xin, Chen, Xiao, & Liu, Qun. 2021. *AutoTinyBERT: Automatic Hyper-parameter Optimization for Efficient Pre-trained Language Models*.

Yu, Licheng, Poirson, Patrick, Yang, Shan, Berg, Alexander C., & Berg, Tamara L. 2016. *Modeling Context in Referring Expressions*.

Zhu, Yukun, Kiros, Ryan, Zemel, Rich, Salakhutdinov, Ruslan, Urtasun, Raquel, Torralba, Antonio, & Fidler, Sanja. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. *Pages 19–27 of: Proceedings of the IEEE international conference on computer vision*.