Visually Grounded Interpretation of Noun-Noun Compounds in English

Abstract

Noun-noun compounds (NNCs) occur frequently in the English language. Accurate NNC interpretation, i.e. determining the implicit relationship between the constituents of a NNC, is crucial for the advancement of many natural language processing tasks. Until now, computational NNC interpretation has been limited to approaches involving linguistic representations only. However, much research suggests that grounding linguistic representations in vision or other modalities can increase performance on this and other tasks. Our work is a novel comparison of linguistic and visuo-linguistic representations for the task of NNC interpretation. We frame NNC interpretation as a relation classification task, evaluating on a large, relationally-annotated NNC dataset. We combine distributional word vectors with image vectors to investigate how visual information can help improve NNC interpretation systems. We find that adding visual vectors increases classification performance on our dataset in many cases.

1 Introduction

Conceptual combination is the cognitive process that allows us to combine two mental concepts into one, for example by juxtaposing or otherwise merging two concepts. For instance, a house located on a beach might typically be called a 'beach house'. Noun-noun compounds (NNCs) are the linguistic phenomenon in which two nouns are joined to form one single, syntactically inseparable unit. The process of combining nouns into new nominal units is both highly prevalent and infinitely productive in a language like English (Libben, 2014), and also exists in various forms in many other languages, including but not limited to German, Norwegian, Hindi, Tamil, Japanese, Chinese, Bulgarian, and Turkish (Nakov, 2013). In English, the *head* of the NNC is usually the rightmost word, and determines the semantic category of the compound. The leftmost word in English NNCs is referred to as the *modifier*. Although NNCs are a common occurrence, the highly productive nature of compounding (Algeo and Algeo, 1993) means that individual NNCs tend to have relatively low frequency counts (Kim and Baldwin, 2006). Compositional models have therefore been of particular interest to researchers working on computational NNC representations (e.g. Shwartz, 2019; Dima, 2016).

Because of the high prevalence and complex nature of English NNCs, the ability of interpret compounds would greatly improve several important natural language processing tasks, such as machine translation (Baldwin and Tanaka, 2004; Balyan and Chatterjee, 2015), text summarization (e.g. Silber and McCoy, 2000), question answering (e.g. Mann, 2002), and natural language inference (e.g. MacCartney and Manning, 2008).

In this paper, we frame compound interpretation as a classification problem, where the goal is to identify the semantic relationship between the nominal elements of a compound. We explicitly compare the contribution of linguistic and multimodal (visuo-linguistic) representations to this task.[1] In part, the motivation for this is theoretical, insofar as a computational account of linguistic meaning has to address the link between symbolic and non-symbolic information (Bender and Koller, 2020; Bisk et al., 2020). A further motivation is the empirical observation that grounding representations in vision gives rise to richer meaning representations (Bruni et al., 2012; Collell Talleda and Moens, 2016). Composition in the visual modality has also been shown to be possible for certain NNCs (Pezzelle et al., 2016). Our goal is to assess to what extent visual grounding helps to accurately

[1] The code for our experiments, as well as our visual embeddings, can be found here: `placeholder`

identify the semantic relationship between NNC constituents. For example, Figure 1 displays images of the constituents of 'beach house' as well as the compound itself. Does the relationship between the constituents in the NNC become easy to predict once such visual information is incorporated, in addition to the textual representation of the constituents?

Figure 1: Picture of a beach, a house, and a beach house from ImageNet.

2 Background

Early approaches to the automatic interpretation of noun compounds included rule-based approaches (Finin, 1980; Vanderwende, 1994) or semi-automatic approaches requiring some user interaction (Barker and Szpakowicz, 1998). Other work utilized frequency statistics of NNC constituents to build probabilistic models for NNC interpretation (Lauer, 1995; Lapata and Keller, 2004). Kim and Baldwin (2005, 2006) leveraged WordNet (Miller, 1998) similarities in supervised training approaches.

Some approaches to NNC interpretation deal with identifying an appropriate paraphrase for a compound which explicitly states the relation between the compound's constituents. Several paraphrasing-based approaches have viewed the task of freely paraphrasing noun compounds as a goal in itself (Hendrickx et al., 2019; Ponkiya et al., 2020; Shwartz and Dagan, 2019; Van de Cruys et al., 2013), whereas others have used paraphrases as inputs to a model, representing NNCs in some way through their paraphrases.

Other approaches to NNC relation classification tend to be centered around classifying NNCs based on a pre-defined set of compound relations using various representations of the compounds themselves as input. Both compositional and distributional representations have been tested. Dima (2016) and Shwartz (2019) both tested various ways of representing noun compounds. Dima (2016) performed the first experiments on compositional representations of English NNCs, using compositional models such as the FullAdd model (Zanzotto et al., 2010) and the Matrix model (Socher et al., 2012). Her results, which were tested on datasets by Tratz and Hovy (2010) and Ó Séaghdha (2008), reached a similar performance to the results obtained by Tratz and Hovy (2010) and Ó Séaghdha (2008), respectively, although their approaches were more complex and involved adding lexical and relational information.

Visuo-linguistic representations for NNC interpretation have received far less attention. Günther et al. (2020) created the first computational model of visuo-linguistic conceptual combination, reporting positive results on several NNC processing tasks. Pezzelle et al. (2016) found that certain compounds can be composed in the visual domain by simple addition of image feature vectors. However, none of these studies have touched upon NNC *interpretation* using visuo-linguistic data, an area that remains unexplored, to our knowledge.

The present work focuses on the interpretation of NNCs that possess at least some degree of compositionality. This is justified on the grounds that novel compounds, which are very common (Algeo and Algeo, 1993), must be interpreted compositionally on first encounter. We employ one compositional model, called the Full Additive model (Zanzotto et al., 2010), as well as simple vector concatenation, in our experiments to construct compound vectors from individual constituent vectors. We do this for linguistic and visual vectors separately, and then combine the two modalities using vector concatenation. The following section will describe how we obtain our visual and linguistic vectors as well as introduce the noun compound dataset that we use.

3 Data

To perform our experiments, we use two main sources of data: a relationally-labeled NNC interpretation dataset by Tratz (2011) to train and test our models on, and ImageNet (Deng et al., 2009) to extract visual feature embeddings. The following subsections will describe these datasets in more detail.

3.1 Compound Dataset

Our main compound dataset for this work is a revised version of the Tratz (2011) noun compound dataset, which contains 19,158 distinct NNCs labeled with 37 fine-grained and 12 coarse-grained relation labels. The dataset is based on a previous one first published by Tratz and Hovy (2010), which

	Split	Train	Val	Test
Coarse	random	13835	928	3701
	lexical (full)	4650	1593	766
	lexical (mod)	9555	5316	3593
	lexical (head)	9048	5516	3900
Fine	random	13968	934	3725
	lexical (full)	4614	1574	843
	lexical (mod)	9511	5270	3846
	lexical (head)	8938	5640	4049

Table 1: Number of samples in each configuration (split and grain) of the Tratz (2011) dataset after our filtering.

contained 17509 compounds categorized by 43 fine-grained constituent relation labels. The compounds were annotated using Amazon's Mechanical Turk service,[2] and used a weighted majority-vote scheme based on ten annotation votes per compound, where Turkers voted on the quality on the other Turkers' decisions in order to even out potential inter-annotator disagreement. On their 43-class annotation task, Tratz and Hovy (2010) report a Cohen's k (Cohen, 1960) of 0.57 as a measure of inter-annotator agreement.

To be able to test how compound interpretation models perform when dealing with unseen constituents, the Tratz (2011) dataset is split in various ways to ensure no previously seen constituents are available in the validation and testing phase. Different *lexical* splits ensure that the test and validation dataset contain no constituents previously seen in the training data – the *lexical mod* split ensures no previously seen modifiers (e.g. 'beach' in 'beach house'), the *lexical head* split ensures no previously seen heads (e.g. 'house'), and the *lexical full* split ensures no previously seen constituents at all. The *random* split does not take into account whether constituents are found in the training data or not.

Before performing our experiments, we do some filtering on the data in which we remove the fine-grained classes PERSONAL_NOUN, PERSONAL_TITLE, and LEXICALIZED. Our reason for removing the PERSONAL_NOUN and PERSONAL_TITLE classes is that there is some doubt as to whether proper names and titles possess the same semantic characteristics as common nouns (Cumming, 2007). Several researchers on noun compound interpretation tasks, no matter which dataset they have used, have decided to remove proper nouns from their data for these very same reasons (e.g. Kim and Baldwin, 2006; Shwartz, 2019). Others, like (Dima and Hinrichs, 2015), have chosen to keep these categories but still acknowledge that their presence in the dataset is questionable. We remove the LEXICALIZED class because our work is mainly centered around how to interpret compounds that have a certain degree of compositionality, seeing as novel compounds, which likely make up the majority of compound types in most corpora, will need to be interpreted compositionally. Table 1 gives an overview of the number of samples in the train, validation, and test sets for each configuration of the Tratz (2011) dataset.

3.2 Image Data

ImageNet (Deng et al., 2009) is a large-scale image database which is structured using the WordNet (Miller, 1998) taxonomy, using synsets to represent sets of word meanings. Since many word classes are difficult to represent visually, ImageNet only contains nouns, and no other lexical categories, from the WordNet hierarchy. ImageNet contains 14,197,122 images, indexed by 21841 synsets[3], which represent different senses of the words.

Selecting Synsets and Images from ImageNet In order to collect the images needed for our task, we have to select all the synsets that were linked to each individual word in our dataset, and then retrieve the image URLs linked to those specific synsets. ImageNet is structured in such a way that one word can be linked to several synsets, and one synset can be linked to several words. Image URLs are associated with specific synsets, not specific words, so to retrieve an image URL from a word, one needs to first select which synset(s) one wants to use to represent that word.

Determining the appropriate sense to use for each constituent in a sample based on their context on the compound level is not trivial. We decide to go for a simple heuristic approach, namely finding the synset that most probably represents the most common or basic meaning for each word, given that the synset has images linked to it (where possible). Our heuristic method consists of the following steps:

1. For a given word, let us call this our *target word*, retrieve all synsets that have images linked to them.

[2] https://www.mturk.com/

[3] As per January of 2022

2. For each of the retrieved synsets, get the list of words that contain that synset among its synsets (representing the potential senses of the word). Let us call this list of words *comparison words*.

3. For each list of *comparison words*, compute the cosine similarity between each *comparison word* and the *target word*, and then take the average of all of these cosine similarities.

4. The synset whose *comparison words* list has the highest cosine similarity to the *target word* is selected as the most common, or basic, meaning.

Note that this method does not necessarily yield the *most common* sense, but the most common *imageable* sense, that is, the most common sense of a word, out of those which have related images. This choice was made on the basis of two assumptions: 1) it would give us the chance to collect more images, as opposed to choosing an approach that selected the most common meaning first and then selected images based on the resulting synsets, and 2) an imageable synset, although perhaps not the most common one related to a certain word, might still have certain visual properties in common with another less imageable, but more common, meaning of said word.

	ImageNet	ResNet10	ResNet100	Total in data
Unique mods	38.7%	36.4%	32.4%	3126
Unique heads	40.1%	37.6%	31.6%	3187

Table 2: Overview of the percentages of unique modifiers and heads in the coarse-grained random split of the (Tratz, 2011) data that have ImageNet images available, and that we could obtain ResNet_{10} and ResNet_{100} vectors for.

Table 2 gives an example of the ImageNet coverage of unique heads and modifiers in one dataset configuration (the random + coarse setting).

4 Methods

We frame the NNC interpretation task as a classification problem, experimenting with passing linguistic and visuo-linguistic vectors as inputs to an SVM classifier. Our experimental process can be described in three steps:

1. Obtain linguistic vectors (from a pre-trained *word2vec* model) and visual feature vectors (from a pre-trained ResNet model) for the constituents of a compound (head and modifier). We experiment with both unimodal (word) embeddings, and visuo-linguistic embeddings, formed by concatenating the word embedding of a compound to the visual representation of a compound.

2. Combine the vector representations of each constituent (either linguistic, or visuo-linguistic).[4] We use two methods of combination, explained below.

3. Observe and evaluate the performance of a setup depending on a) mode of constituent vector combination and b) modality of vectors (purely linguistic, or visuo-linguistic)

Given the vector representations (unimodal/linguistic, or bimodal/visuo-linguistic) for the two constituents of a compound, we experiment with two methods of combination to obtain a representation for the compound as a whole: (a) simple concatenation; and (b) the Full Additive (FullAdd) method proposed by Zanzotto et al. (2010). To obtain linguistic vectors and visual vectors, we utilize pre-trained word2vec (Mikolov et al., 2013a) and ResNet (He et al., 2016) models, respectively. All of these models, as well as our experimental setups and baselines, will be described in this section.

4.1 Models

This section will give an overview of the models we use for obtaining feature vectors as well as composing said vectors into unimodal and bimodal compound representations.

4.1.1 The Full Additive Model

The Full Additive model, also referred to as FullAdd or the Estimated Additive model (Zanzotto et al., 2010) is a model where the two vectors $\overrightarrow{x}$ and $\overrightarrow{y}$, representing the constituent words c_1 and c_2, are multiplied by square matrices A and B, respectively, and then added together to create a compositional meaning representation of a phrase. A and B are the same for each vector $\overrightarrow{x}$ and $\overrightarrow{y}$, respectively, and are obtained through training on a training set of compound nouns that contains distributional vector representations of each compound and each constituent word. We can think

[4] In case a constituent lacks a vector representation in either modality, we instead use a vector of zeros.

of these vectors as being ordered in triples, where any triple of words *(z, x, y)*, which corresponds to *(compound, modifier, head)* in English, is represented by a triple of vectors $(\overrightarrow{z}, \overrightarrow{x}, \overrightarrow{y})$. For example, the training set could contain the vector triple $(\overrightarrow{soap\ opera}, \overrightarrow{soap}, \overrightarrow{opera})$. The goal will be to learn a composition function for any word vectors $\overrightarrow{x}, \overrightarrow{y}$ such that $\overrightarrow{p} = f(\overrightarrow{x}, \overrightarrow{y})$ approximates $\overrightarrow{z}$, where $\overrightarrow{p}$ is the composed vector for any given noun compound, and $\overrightarrow{z}$ is the observed distributional vector for this noun compound. Intuitively, one can think of the process of training the two matrices (one for modifiers and one for heads) as finding a way of transforming a meaning representation of a single word into its *as-constituent* meaning. For example, by multiplying the vector for *chocolate* with the modifier matrix (which we call matrix *A*), we approximate the *as-modifer* meaning of *chocolate*, as in *chocolate cake*. The general equation for composing a compound vector $\overrightarrow{z}$ given two constituent word vectors $\overrightarrow{x}$ and $\overrightarrow{y}$ is given below:

$$\overrightarrow{z} = \mathbf{A}\overrightarrow{x} + \mathbf{B}\overrightarrow{y} \tag{1}$$

To implement our FullAdd model, we use the Distributional Semantic Composition Toolkit, or DISSECT (Dinu et al., 2013), which allows for the implementation of FullAdd as well as other composition models. To prepare the necessary data for FullAdd, we filter our training data so that we only keep the compounds for which the whole compound as well as the modifier and head separately have vectors associated with them in our word embedding model. Then we construct a semantic space using those word embeddings. Because of the requirements of the DISSECT implementation, heads and modifiers cannot be repeated in the semantic space (e.g., we can not include both 'cat food' and 'dog food'). The two FullAdd matrices, **A** and **B**, can then be trained in the way described above. The resulting vectors are then used to compose compositional meaning vectors for our training, test, and validation data. In our FullAdd experiments, we train a FullAdd model for each modality (linguistic and visual) and then create composed vectors for each compound in each modality before combining the two modalities using concatenation.

4.1.2 ResNet

ResNet (He et al., 2016) is a deep residual neural network architecture for image recognition. ResNet models learn residual functions instead of unreferenced functions, allowing for the creation of models that are deeper than previous CNN models such as the VGG models (Chatfield et al., 2014), while still being less complex and faster to train (He et al., 2016). The ResNet152 model, which at the time of its release was the deepest neural network trained on ImageNet (He et al., 2016), with its 152 layers, won the ImageNet Large Scale Visual Recognition Challenge in 2015 (Russakovsky et al., 2015). To extract visual embeddings based on images from ImageNet, we use a ResNet152 model trained on ImageNet data, implemented in the Keras (Chollet et al., 2015) library for Python. ResNet is trained on an object classification task, using 1.28 million images in its training phase. The model learns to take an image vector as input and outputs one out of the 1000 ImageNet category labels included in its training data.

To extract visual features using ResNet152, we flatten the final layer before the final classification (softmax) layer of the model, which has the size 7 x 7 x 2048, resulting in vectors of size 100352. We use two experimental settings for visual features, where we extract and average feature vectors for 10 or 100 ImageNet images. We will refer to these vectors as ResNet_{10} and ResNet_{100}, respectively. See Table 2 for a summary of the image availability in the Tratz (2011) dataset. These vectors can then be reduced to our desired vector dimensions, for example 300 in order for them to be compatible with pre-trained 300-dimensional *word2vec* embeddings. For our ResNet vectors to be more appropriate as inputs to our SVM classifiers, we scale our vectors so that the values range from -1 to 1.

4.1.3 word2vec

To obtain word embeddings to use as our linguistic vectors, we use a pre-trained word2vec model (Mikolov et al., 2013a). We employ a popular set of pre-trained word2vec embeddings that were trained on about 100 billion words from the GoogleNews dataset. These 300-dimensional word embeddings[5] were trained using a skip-gram approach with negative sampling (SGNS for short), as described in

[5] https://code.google.com/archive/p/word2vec/

Mikolov et al. (2013b). Unlike previous work on this dataset published by e.g. Shwartz (2019), we decide to not train our own word2vec embeddings. This decision was made because our goal is investigating the effect of combining linguistic representations with visual ones, rather than comparing different kinds of linguistic representations, like Shwartz (2019) did.

4.2 Experimental Setups

For our experiments, we create three majority-class baselines in addition to our SVM classifier.[6] In this section, we will describe our baselines and our main experimental setups.

4.2.1 Baselines

We implement the following majority-class baselines:

- Overall Majority: For a given data sample, this baseline selects the overall majority class as observed in the training data.
- Modifier Majority: For a given data sample, this baseline selects the majority class represented among compounds in the training data with the same modifier as the sample.
- Head Majority: For a given data sample, this baseline selects the majority class represented among compounds in the training data with the same head as the sample.

The 'Modifier Majority' and 'Head Majority' rely on using the modifiers or heads, respectively, from the training data to determine the assigned label of each data sample. However, we have several dataset configurations in which the training and test datasets do not share any heads, modifiers, or any constituents at all – see Table 1 for a summary. Thus, in these configurations, the head or modifier majority mechanism will not work. We select the overall majority in those cases. This means that for the lexical + mod split of our data, the Modifier Majority baseline will give the exact same results as the Overall Majority baseline. The same is the case for the lexical + head split together with the Head Majority baseline, as well as the full lexical split with both the Modifier Majority and Head Majority baselines.

[6]We did also perform a few NNC interpretation experiments using a BERT model, which were not included in this paper because of poor performance on the lexical splits of the Tratz (2011) dataset. See Table 7 in the appendix for an overview.

4.2.2 Classifier Setup

Our classifier is an SVM that takes as inputs either linguistic representations (in the form of *word2vec* vectors that have either been concatenated or composed using the FullAdd function) or visuo-linguistic representations (in the form of linguistic vectors concatenated with the visual vectors described in section 4.1.2). For our SVM we use a linear kernel, L2 penalty and a C value of 0.5. We train our classifier on the Tratz (2011) data, passing either our linguistic or visuo-linguistic vectors as inputs.

5 Results and Evaluation

We evaluate our setup on the Tratz (2011) dataset and report F1 scores for all dataset configurations.

	Split	**MC-O**	**MC-M**	**MC-H**
Coarse	random	7.5	40.0	59.3
	lexical (full)	6.7	–	–
	lexical (mod)	7.8	–	58.8
	lexical (head)	7.0	38.6	–
Fine	random	5.3	34.5	54.1
	lexical (full)	5.6	–	–
	lexical (mod)	6.3	–	52.6
	lexical (head)	5.2	34.8	–

Table 3: F1 scores from our baseline classifiers

Table 3 shows the weighted F1 scores of our baseline classifiers. The modifier- and head-majority classifiers require the test datasets to include previously seen modifiers and heads (respectively), which is why the table has some cells that are marked with '–', indicating that the score for this cannot be computed with the given majority-class strategy and thus would get the same score as the overall majority baseline. For this reason, we only have comparable scores from the modifier- and head-majority classifiers in the case of the random split, in which both the fine-grained setting and the coarse-grained setting show that the head-majority classifier performs the best. In other words, it seems that having a common head is a greater indicator of same-class membership than having a common modifier in the Tratz (2011) dataset.

Table 4 shows the results of our experiments on the Tratz (2011) after our filtering. All of the scores given in the tables are F1 scores, and an asterisk next to an increase in score means that the increase was found to be significant following a McNemar test and a Bonferroni correction of the

	Split	word2vec	w2v + ResNet10		w2v + ResNet100	
Coarse	random	66.3	66.0	*- 0.3*	**66.4**	*+ 0.1*
	lexical (full)	**44.2**	44.1	*- 0.1*	43.7	*- 0.5*
	lexical (mod)	57.9	**58.3**	*+ 0.4*	57.7	*- 0.2*
	lexical (head)	50.8	51.0	*+ 0.2*	**51.3**	*+ 0.5*
Fine	random	**66.7**	66.6	*- 0.1*	**66.7**	*+/- 0*
	lexical (full)	39.2	**39.4**	*+ 0.2*	38.4	*- 0.8*
	lexical (mod)	56.4	56.4	*+/- 0*	**56.5**	*+ 0.1*
	lexical (head)	47.1	**47.5**	*+ 0.4*	46.9	*- 0.2*

(a) F1 scores using FullAdd-composed compound vectors

	Split	word2vec	w2v + ResNet10		w2v + ResNet100	
Coarse	random	74.1	**75.3**	*+ 1.2**	75.2	*+ 1.1**
	lexical (full)	49.1	**50.8**	*+ 1.7**	50.0	*+ 0.9*
	lexical (mod)	63.5	**64.0**	*+ 0.5*	63.1	*- 0.4*
	lexical (head)	55.5	**56.7**	*+ 1.2*	56.0	*+ 0.5*
Fine	random	73.0	**75.0**	*+ 2.0*	**75.0**	*+ 2.0*
	lexical (full)	40.3	40.0	*- 0.3*	**40.7**	*+ 0.4*
	lexical (mod)	63.0	**63.5**	*+ 0.5*	63.4	*+ 0.4*
	lexical (head)	50.6	51.6	*+ 1.0**	**52.0**	*+ 1.4**

(b) Results using concatenated compound vectors

Table 4: Weighted F1 scores from classification experiments using linguistic and visuo-linguistic vectors. The tables show results of using FullAdd-composed vectors as well as concatenation-composed vectors, with the change in F1 obtained when ResNet vectors are included. An asterisk next to an increase in F1 score means the bimodal result is significantly different from its unimodal counterpart following a Bonferroni-corrected McNemar test.

p-values.[7] As is evident when comparing tables 4a and 4b, using concatenated vectors as opposed to FullAdd composed vectors yields much higher F1 scores. Additionally, the results in table 4b, with concatenated vectors, are less ambiguous: in this experiment, at least one of the visuo-linguistic settings beats the purely linguistic setting in each experimental setting.

As has been shown in previous research on this dataset, the most challenging dataset split is the full lexical split, where no constituents in the validation and test data are previously seen in the training data. As expected, the fine-grained setting is generally more of a challenge than the coarse-grained one. As we can see from comparing tables 4a and 4b, the results in the former are more ambiguous, meaning that we cannot conclude that one input type (linguistic or visuo-linguistic vectors) is better than another. In table 4b, however, we find that our visuo-linguistic vectors help increase scores significantly in a few cases. In the case of the ResNet_{10} vectors, the increase in scores is significant for the random and full lexical splits in the coarse-grained setting as well as for the lexical (head) split in the fine-grained setting. For our ResNet_{100} vectors, only the coarse + random and the fine + lexical (head) settings show a significant increase in scores.

One might be inclined to question why our FullAdd experiments on this dataset perform worse than very similar experiments done by e.g. Shwartz (2019) and Dima (2016). This is likely due to the fact that Shwartz (2019) and Dima (2016) trained their own word embeddings specifically for this task, meaning that they were able to obtain distributed embeddings for more of the compounds in the Tratz (2011) dataset than what we had available through our pre-trained model, and as a consequence had more training data for the FullAdd model. As our goal with this work is not to compare composition functions for linguistic vectors, we saw training our own embeddings as being superfluous for this study.

Overall, we see that, in our experiments with concatenated compound vectors, adding visual information helps increase the scores in all cases, and in some cases the increases are statistically significant.

5.1 Concreteness Ratings

Intuitively, one could assume that visual information (i.e. images) would be easier to obtain for more concrete words, thus making visual information a more appropriate and/or helpful addition for compounds that have relatively concrete constituents. If this is the case, then we should find a higher benefit of incorporating visual information, the more concrete a word is.

We quantify concreteness using a dataset of concreteness ratings of almost 40,000 English lemmas, by Brysbaert et al. (2014). The ratings are continuous values between 1 and 5, where 5 is the most concrete. The ratings were obtained by surveying more than 4,000 participants in a crowdsourcing study and taking the mean of the ratings obtained for each word. By hypothesis, the more concrete a word is, the more images related to that word should be available, and therefore, the greater the benefit of incorporating visual features in the representation of a compound containing that word.

As a first analysis of our results in light of these concreteness ratings, we performed several logistic regression analyses where we looked at the concreteness ratings of modifiers and heads as predictors of classification success. Table 6 in Appendix A gives a full overview of these results. What we

[7] We set our α level to the conventional 0.05, which resulted in a p-value threshold of 0.00625 after a Bonferroni correction.

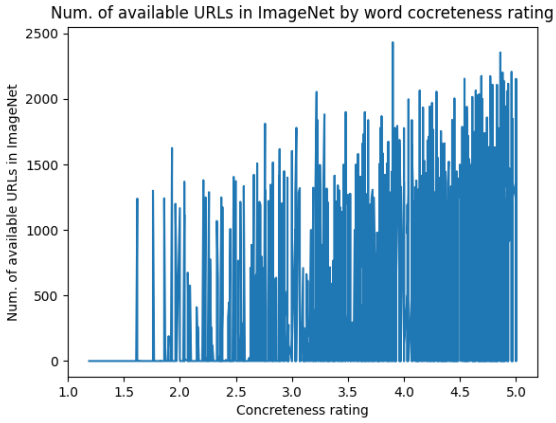

Figure 2: Word concreteness rating by number of available URLs in ImageNet

find is that the dataset configuration seems to matter more than the modality, but that the concreteness ratings of both modifiers and heads are, in some cases, good predictors of classifier success. However, in the significant cases, we discover a negative relationship between concreteness and classification success – i.e., the higher the concreteness of a modifier/head, the lower the chance of the classifier predicting the correct class. To investigate this more, we performed a detailed investigation of some of our results, sorting the samples by image availability (specifically, whether a constituent had fewer or more than 10 images available in ImageNet). The full results are found in Table 5 in Appendix A.

Figure 2 shows word concreteness ratings by number of URLs available in ImageNet, as determined by our image selection heuristic, for each of the words in the Tratz (2011) dataset that had concreteness ratings in the Brysbaert et al. (2014) dataset (regardless of whether they appeared as a modifier or head).

A correlation analysis revealed a low to moderate correlation between the concreteness ratings and the URL counts (Pearson's $r = 0.45$, $p < 0.001$). This indicates that, to some extent, the higher the concreteness rating of a constituent in a compound, the higher the chances of finding 10 or 100 images to represent said constituent as part of our image vectors. Examples of words for which we were not able to obtain at least 10 images include *minute* (concreteness rating 3.04), *intelligence* (concreteness rating 2.24), and *state* (concreteness rating 3.52).

The negative relationship between constituent concreteness and classifier success seems counter-intuitive, but might be a results of a number of factors related to word frequency, polysemy, and the distribution of concrete vs. non-concrete words over the classes in the Tratz (2011) dataset. We leave it up to future work to investigate these relationships, but we note that our visuo-linguistic representations do tend to outperform the purely linguistic ones, regardless of constituent concreteness ratings.

6 Conclusion and Future Work

In this paper, we have presented NNC interpretation experiments on the Tratz (2011) dataset, comparing linguistic and visuo-linguistic inputs to an SVM classifier. We have found that, in our best-performing case, combining visual feature vectors with linguistic vectors (word embeddings) helps increase F1 scores on the Tratz (2011) dataset in almost all experimental settings. Our findings indicate that utilizing visual information for this NNC relation classification task is indeed a promising endeavor.

Future work should aim to further refine our approach by for example using more sophisticated methods for selecting images to represent words, exploring ways to represent abstract or non-imageable words, and finding better ways to visually ground polysemous words. In this regard, recent multimodal encoders pretrained on visual and linguistic data (e.g. Lu et al., 2019; Tan and Bansal, 2019), are a promising way forward. Another possible angle for future work could be to consider NNC interpretation in visual and linguistic contexts. In the future, we would also be eager to explore visual grounding in other aspects of computational NNC related tasks, such as NNC generation. Additionally, our approach should be tested on different datasets and in different circumstances, for example in a task that determines the probability of compound categories rather than fixed classes. One final potential angle for future work could be to look further into the task of visual composition. A first step could be to more closely examine the effects of using the FullAdd function with image vectors.

To conclude, our results are in line with previous work from both cognitive science and computational linguistics suggesting that more psychologically plausible models of NNC processing should incorporate grounding.

References

John Algeo and Adele S Algeo. 1993. *Fifty years among the new words: A dictionary of neologisms 1941-1991*. Cambridge University Press.

Timothy Baldwin and Takaaki Tanaka. 2004. Translation by machine of complex nominals: Getting it right. In *Proceedings of the Workshop on Multiword Expressions: Integrating Processing*, pages 24–31.

Renu Balyan and Niladri Chatterjee. 2015. Translating noun compounds using semantic relations. *Computer Speech & Language*, 32(1):91–108.

Ken Barker and Stan Szpakowicz. 1998. Semi-automatic recognition of noun modifier relationships. In *COLING 1998 Volume 1: The 17th International Conference on Computational Linguistics*.

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198, Online. Association for Computational Linguistics.

Yonatan Bisk, Ari Holtzman, Jesse Thomason, Jacob Andreas, Yoshua Bengio, Joyce Chai, Mirella Lapata, Angeliki Lazaridou, Jonathan May, Aleksandr Nisnevich, et al. 2020. Experience grounds language. *arXiv preprint arXiv:2004.10151*.

Elia Bruni, Gemma Boleda, Marco Baroni, and Nam-Khanh Tran. 2012. Distributional semantics in technicolor. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 136–145.

Marc Brysbaert, Amy Beth Warriner, and Victor Kuperman. 2014. Concreteness ratings for 40 thousand generally known english word lemmas. *Behavior research methods*, 46(3):904–911.

Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*.

Francois Chollet et al. 2015. Keras.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46.

Guillem Collell Talleda and Marie-Francine Moens. 2016. Is an image worth more than a thousand words? on the fine-grain semantic differences between visual and linguistic representations. In *Proceedings of the 26th International Conference on Computational Linguistics*, pages 2807–2817. ACL.

Tim Van de Cruys, Stergos Afantenos, and Philippe Muller. 2013. Melodi: A supervised distributional approach for free paraphrasing of noun compounds. In *7th International Workshop on Semantic Evaluation (SemEval 2013) in: 2nd Joint Conference on Lexical and Computational Semantics (SEM 2013)*, pages pp–144.

Samuel John Cumming. 2007. *Proper nouns*. Rutgers The State University of New Jersey-New Brunswick.

J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Corina Dima. 2016. On the compositionality and semantic interpretation of english noun compounds. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 27–39.

Corina Dima and Erhard Hinrichs. 2015. Automatic noun compound interpretation using deep neural networks and word embeddings. In *Proceedings of the 11th International Conference on Computational Semantics*, pages 173–183.

Georgiana Dinu, Marco Baroni, et al. 2013. Dissect - distributional semantics composition toolkit. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 31–36.

Rotem Dror, Segev Shlomov, and Roi Reichart. 2019. Deep dominance-how to properly compare deep neural models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2773–2785.

Timothy Wilking Finin. 1980. *The semantic interpretation of compound nominals*. University of Illinois at Urbana-Champaign.

Fritz Günther, Marco Alessandro Petilli, and Marco Marelli. 2020. Semantic transparency is not invisibility: A computational model of perceptually-grounded conceptual combination in word processing. *Journal of Memory and Language*, 112:104104.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Iris Hendrickx, Preslav Nakov, Stan Szpakowicz, Zornitsa Kozareva, Diarmuid O Séaghdha, and Tony Veale. 2019. Semeval-2013 task 4: Free paraphrases of noun compounds. *arXiv preprint arXiv:1911.10421*.

Su Nam Kim and Timothy Baldwin. 2005. Automatic interpretation of noun compounds using wordnet similarity. In *International Conference on Natural Language Processing*, pages 945–956. Springer.

Su Nam Kim and Timothy Baldwin. 2006. Interpreting semantic relations in noun compounds via verb semantics. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 491–498.

Mirella Lapata and Frank Keller. 2004. The web as a baseline: Evaluating the performance of unsupervised web-based models for a range of nlp tasks. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 121–128.

Mark Lauer. 1995. *Designing statistical language learners: Experiments on compound nouns*. Ph.D. thesis, Ph. D. thesis, Macquarie University.

Gary Libben. 2014. The nature of compounds: A psychocentric perspective. *Cognitive neuropsychology*, 31(1-2):8–25.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *arXiv preprint arXiv:1908.02265*.

Bill MacCartney and Christopher D Manning. 2008. Modeling semantic containment and exclusion in natural language inference. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 521–528.

Gideon Mann. 2002. Fine-grained proper noun ontologies for question answering. In *COLING-02: SEMANET: Building and Using Semantic Networks*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

George A Miller. 1998. *WordNet: An electronic lexical database*. MIT press.

Preslav Nakov. 2013. On the interpretation of noun compounds: Syntax, semantics, and entailment. *Natural Language Engineering*, 19(3):291–330.

Diarmuid Ó Séaghdha. 2008. Learning compound noun semantics. Technical report, University of Cambridge, Computer Laboratory.

Sandro Pezzelle, Ravi Shekhar, and Raffaella Bernardi. 2016. Building a bagpipe with a bag and a pipe: Exploring conceptual combination in vision. In *Proceedings of the 5th Workshop on Vision and Language*, pages 60–64.

Girishkumar Ponkiya, Rudra Murthy, Pushpak Bhattacharyya, and Girish Palshikar. 2020. Looking inside noun compounds: Unsupervised prepositional and free paraphrasing using language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 4313–4323.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252.

Vered Shwartz. 2019. A systematic comparison of english noun compound representations. *arXiv preprint arXiv:1906.04772*.

Vered Shwartz and Ido Dagan. 2019. Still a pain in the neck: Evaluating text representations on lexical composition. *Transactions of the Association for Computational Linguistics*, 7:403–419.

H Gregory Silber and Kathleen F McCoy. 2000. Efficient text summarization using lexical chains. In *Proceedings of the 5th international conference on Intelligent user interfaces*, pages 252–255.

Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 1201–1211.

Hao Tan and Mohit Bansal. 2019. LXMERT : Learning Cross-Modality Encoder Representations from Transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processingand the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP'19)*, pages 5100–5111, Hong Kong, China. Association for Computational Linguistics.

Stephen Tratz. 2011. *Semantically-enriched parsing for natural language understanding*. University of Southern California.

Stephen Tratz and Eduard Hovy. 2010. A Taxonomy, Dataset, and Classifier for Automatic Noun Compound Interpretation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 678–687, Uppsala, Sweden. Association for Computational Linguistics.

Lucy Vanderwende. 1994. Algorithm for automatic interpretation of noun sequences. In *COLING 1994 Volume 2: The 15th International Conference on Computational Linguistics*.

Fabio Massimo Zanzotto, Ioannis Korkontzelos, Francesca Fallucchi, and Suresh Manandhar. 2010.

Estimating linear models for compositional distributional semantics. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1263–1271.

Appendix

A Detailed Results Tables

This appendix contains supplementary tables that describe some of our findings in more detail. Table 5 is an overview of the results of our classification algorithm when used on linguistic (L) and visuo-linguistic (VL) vectors. The table shows the F1 scores for subsets of our test data, where we select data samples where either one, both, or none of the constituents in each sample had a ResNet_{10} vector available (i.e., had 10 or more images available in ImageNet). The 'no filtering' column contains the exact same results, for the full dataset, as reported in our main article, and is included for comparison.

	constituents with 10+ images:	*no filtering*		*mods*		*heads*		*both*		*none*	
	split	**L**	**VL**	**L**	**VL**	**L**	**VL**	**L**	**VL**	**L**	**VL**
Coarse	random	74.1	**75.3***	70.46	71.43	72.78	74.65	70.76	72.59	76.84	77.9
	lexical (full)	49.1	**50.8***	41.29	46.04	48.4	51.24	41.3	43.22	56.84	53.77
	lexical (mod)	63.5	64.0	56.77	57.25	59.36	61.35	54.48	55.45	70.22	69.69
	lexical (head)	55.5	56.7	54.3	55.7	50.85	**54.64***	51.42	**55.96***	58.99	59.14
Fine	random	73.0	75.0	69.49	**71.72***	69.19	**72.06***	66.13	69.43	76.09	**77.78***
	lexical (full)	40.3	40.0	41.19	40.91	39.21	35.04	42.03	38.94	42.33	44.59
	lexical (mod)	63.0	63.5	60.13	60.2	54.32	55.77	51.88	53.38	68.71	69.35
	lexical (head)	50.6	**51.6***	52.06	53.35	51.18	51.6	52.88	53.91	50.17	51.53

Table 5: Results of our experiments using the concatenation method of composition and the ResNet_{10} vectors, filtered by the imageability (as modeled by whether or not 10 or more images were available) of the constituents constituents. An asterisk next to a VL score means that the visuo-linguistic (VL) modality performed significantly better than the linguistic (L) modality following a McNemar test with a Bonferroni correction of the p-values.

Table 6 contains a summary of several logistic regression analyses performed on our classification results in both the linguistic and visuo-linguistic modalities. The results show coefficients and p-values of analyses using modifier and head concreteness (separately) as predictors of classification success.

		modifiers				*heads*			
		L		*VL*		*L*		*VL*	
split		**coef**	**p**	**coef**	**p**	**coef**	**p**	**coef**	**p**
Coarse	random	**-0.2049**	**<0.001**	**-0.1675**	**<0.001**	-0.18	0.008	-0.99	0.028
	lexical (full)	-0.1788	0.032	-0.1251	0.133	0.076	0.352	0.13	0.113
	lexical (mod)	**-0.2322**	**<0.001**	**-0.2740**	**<0.001**	**-0.1924**	**<0.001**	**-0.2093**	**<0.001**
	lexical (head)	**-0.1710**	**<0.001**	**-0.2118**	**<0.001**	0.0053	0.888	0.0432	0.252
Fine	random	**-0.2128**	**<0.001**	**-0.2073**	**<0.001**	**-0.268**	**<0.001**	**-0.27**	**<0.001**
	lexical (full)	0.1108	0.171	0.1665	0.041	0.0854	0.340	0.0296	0.741
	lexical (mod)	**-0.1340**	**0.001**	**-0.1581**	**<0.001**	**-0.3257**	**<0.001**	**-0.3306**	**<0.001**
	lexical (head)	0.0393	0.278	0.0423	0.243	0.0659	0.090	0.0569	0.144

Table 6: Results from a logistic regression analysis of modifier and head concreteness as a predictor of the successful classification of compounds. The scores in boldface are ones where the p-values are lower than a Bonferroni-corrected α level of 0.05.

Table 7 shows the results of some NNC interpretation experiments that we did with a pre-trained BERT model (Devlin et al., 2018) and our ResNet visual embeddings. In these experiments, we fed compounds to a BERT model fitted with a linear classifier on top in order to get the classifications of the compounds.

split	grain	BERT	BERT + ResNet$_{10_RAW}$			BERT + ResNet$_{10_NORM}$		
			F1	diff	$\epsilon, (B<BM_{RAW})$	F1	diff	$\epsilon, (B<BM_{NORM})$
random	coarse	**78.7**	69.7	- *9*	0.99	**78.7**	+/- *0*	0.47*
random	fine	57.9	50.7	- *7.2*	0.94	**65.1**	+ *7.2*	0.024*
lexical (full)	coarse	**31.6**	28.6	- *3*	0.92	25.8	- *5.8*	0.95
lexical (full)	fine	**19.5**	14.5	- *5*	1.00	15.0	- *4.5*	0.79
lexical (mod)	coarse	17.0	**36.6**	+ *19.6*	0.036*	6.7	- *10.3*	0.98
lexical (mod)	fine	8.3	**27.0**	+ *18.7*	0.005*	8.3	+/- *0*	0.55
lexical (head)	coarse	11.1	**40.2**	+ *29.1*	0.004*	5.0	- *6.1*	0.84
lexical (head)	fine	4.4	**29.5**	+ *25.1*	0.036*	2.8	- *1.6*	0.78

Table 7: Results from fine-tuning BERT with and without adding ResNet$_{10}$ vectors after 50 epochs of training, averaged over 10 runs. Each column of bimodal results shows weighted F1, the change in F1 between the unimodal and the given bimodal setting, and the epsilon value from the ASD algorithm that reveals to what extent the bimodal is better than the unimodal setting.

In the visuo-linguistic modality, we concatenated BERT's linguistic embeddings with our visual embeddings before passing them to a linear classification layer. We experimented with using raw ResNet embeddings (straight out of the ResNet model, without applying anything but dimensionality reduction) and normalized ResNet embeddings. The table shows F1 scores as well as the ϵ value returned by the *Almost Stochastic Dominance* (ASD) algorithm proposed by Dror et al. (2019) for comparing the performance of two neural network architectures. The algorithm works in such a way that an ϵ value of less than 0.5 means that algorithm B (in our case, one of the visuo-linguistic settings) is *almost stochastically dominant* over algorithm A (in our case, the purely linguistic setting).