Definitional Templating: A Novel Approach to Modeling the Compositional Semantics of Noun Compounds

Adam Davies
University of Utah

UUCS-21-013

School of Computing
University of Utah
Salt Lake City, UT 84112 USA

6 May 2021

Abstract

Interpreting the meaning of noun compounds (e.g., party member or bus stop) is an ongoing challenge in natural language processing (NLP). Much of this difficulty arises from the fact that the relationship between nouns in a compound is largely implicit, so interpreting this relationship often requires extrapolation beyond the meanings of the nouns themselves. I have explored a novel approach to representing noun compound relations, denitional templating, which explicates their meanings in comprehensible terms for NLP systems. This approach is shown to improve performance in interpreting the relationship between constituents of a noun compound across multiple experimental contexts, relative to analogous approaches which represent only the nouns themselves.

Definitional Templating: A Novel Approach to Modeling the Compositional Semantics of Noun Compounds

by

Adam Davies

A Senior Thesis Submitted to the Faculty of
The University of Utah
In Partial Fulfillment of the Requirements for the

Degree of Bachelor of Science

In

Computer Science

Approved:

[illegible]

Ellen Riloff
Thesis Faculty Supervisor

Mary Hall
Director, School of Computing

Jim de St. Germain
Director of Undergraduate Studies

May 2021

Abstract

Interpreting the meaning of noun compounds (e.g., *party member* or *bus stop*) is an ongoing challenge in natural language processing (NLP). Much of this difficulty arises from the fact that the relationship between nouns in a compound is largely implicit, so interpreting this relationship often requires extrapolation beyond the meanings of the nouns themselves. I have explored a novel approach to representing noun compound relations, *definitional templating*, which explicates their meanings in comprehensible terms for NLP systems. This approach is shown to improve performance in interpreting the relationship between constituents of a noun compound across multiple experimental contexts, relative to analogous approaches which represent only the nouns themselves.

Contents

1 Introduction

Noun compound interpretation is, as one might expect, the task of interpreting the meanings of noun compounds (NCs, for short), which are sequences of two or more nouns where one noun (called the *head*) is modified by the other. For example, in the NC *university professor*, it is understood that *university* is a modifier of the head, *professor*, where the employer and/or workplace of the *professor* is clarified by the modifier, *university.*

There are two levels at which one may analyze the meaning of an NC. The first is that of *lexical semantics*, where the meaning of an NC is understood at the level of each noun individually. The second is the level of *compositional semantics*, where the meaning of an NC is understood in terms of the (implicit) relationship between its constituents, determining *how* the modifier modifies the head. In these terms, the task of NC interpretation can be understood as determining the compositional semantics of an NC (i.e., the nature of the relationship between the nouns in the compound) on the basis of the lexical semantics of an NC (which may be known on the basis of the words alone). Both levels of semantics are essential to NC interpretation, although the question of whether they must both be explicitly represented remains an open question.

One common formulation of NC interpretation is *noun compound relation classification* (*NC classification*, for short), where machine learning-based classifiers are trained to select the correct relation for an NC from a fixed, predetermined set of relation types that classify the relationship between its constituents. For example, the relation class of *university professor* could be labeled as *Employer*, to indicate that the *university* is the employer of the *professor*, whereas *silicon chip* might be labeled as *Containment*, to indicate that the *chip* contains *silicon.* (These are the relation labels assigned to *university professor* and *silicon chip* by Tratz 2011, whose approach to NC categorization is discussed in §2.1.)

This research explores a novel approach to representing relation classes in NC classification: *definitional templating*, which models the meaning of each relation as a collection of one or more *templates*. Each template expresses the definition of its corresponding relation in natural language, and has two slots which may be filled by the constituents of a 2-word NC. For example, the template developed for the relation exhibited by *silicon chip*, *Containment*, is:

The *N2* is made of *N1*.

Which, filled with the NC *silicon chip*, is:

The *chip* is made of *silicon*.

In general terms, statistical language models (LMs) are trained to predict what an unknown word in a sequence of words is likely to be, given the other words in that sequence.[1] This research applies a recent, high-performing LM, BERT (Devlin et al. 2018), in determining how likely two words (an NC's constituents) are to fit in a given sequence (a template). My hypothesis is that LMs such as BERT (Devlin et al.) may be able to consistently distinguish between sentences containing one or more words which are out-of-place in the semantic context of the sentence – in this case, a template containing the constituents of an NC associated with a different relation than the template – and sentences which do not contain such out-of-context words. For example, an LM may be able to distinguish how contextually appropriate *silicon chip* is in the context of the following two templates (which, as we will see in §3.3, are those for the *Containment* and *Time* relations):

The *chip* is made of *silicon*.

The *chip* occurs during *silicon*.*

If the LM is able to make such distinctions (e.g., by determining that *silicon chip* is more probable in the context of the first template than the second), then it may be possible to leverage these distinctions to improve performance in NC classification. This is the principle behind the definitional templating approach: a variety of NC classifiers (all of which are based on BERT, Devlin et al. 2018) are trained to map filled templates as inputs to binary outputs corresponding to whether the NC in a filled template matches the relation in that template or not. Once a classifier is trained, it classifies an NC as having the relation described by the template which the classifier determines that the NC best matches.

2 Related Work

2.1 NC Categorization

The semantics of NCs have long been a productive research area in linguistics. Of particular interest in my research is the work which has focused on categorizing NCs on the basis of the relationship between their constituents. A few examples of such categorization schemes, and the justification behind them, are outlined below.

One influential perspective on the categorization of NCs has been the *generative approach*, which understands NCs as being derived forms of larger syntactic structures containing the NC's constituents (see Lees 1960, Levi 1978). The most prominent

[1] Traditionally, language modeling is formulated in terms of predicting the next word in a sequence on the basis of all preceding words; but the LM used in this work, BERT (Devlin et al. 2018), is trained to predict missing words at any point in a sequence, and can make use of words before *and* after the missing word to predict it.

formulation of this perspective is likely that of Levi, who argued that all NCs (with the exception of nominalized NCs) are derived from a small number of predicates (specifically, *Cause*, *Have*, *Make*, *Be*, *Use*, *For*, *In*, *About*, *From*) containing the NC constituents. For example, the *Make* predicate follows the form *N2 which makes N1*, meaning that an NC like *honey bee* is derived from the predicate *bee which makes honey*. NCs are obtained by simply deleting the predicate, and may be categorized according to the type of predicate from which they were formed.

Others have argued that NCs cannot be neatly packaged into a small, closed set of categories. According to Downing 1977, "attempts to characterize [noun] compounds as derived from a limited set of [semantic or syntactic] structures can only be considered misguided," because NCs are freely constructed, ad-hoc, to describe diverse semantic relationships between their constituents. On this view, classifying NCs according to a limited set of relationships – in essence, the task of NC classification, as described in §1 – is an inadequate proxy for NC interpretation.

Naturally, varying perspectives on the categorization of NCs lead to different ideas about how they should be categorized. Top-down theoretical perspectives like the *generative approach* have led many researchers to claim that all (or nearly all) NCs may be sorted into a limited number of well-defined groups of NCs characterized by common syntactic or semantic origins, which has led them to create their own such models of NC categorization (including Levi 1978, Warren 1978, Girju et al. 2005, Ó Séaghdha 2008, Tratz and Hovy 2010, and Tratz 2011). For example, Tratz 2011 established a taxonomy of 37 fine-grained relation types, each of which is a sub-category among a set of 11 coarse-grained relation categories.[2] Following this taxonomy, he created and published a dataset with ∼19k English NCs, each with a label corresponding to the coarse- and fine-grained relation it exhibits. This is the dataset which I have used throughout my research (see §3.2 for more details), but the templating approach could straightforwardly be extended to other datasets by applying methods similar to those which I describe in §3.3.

A reasonable response to criticisms like Downing 1977's is to consider alternative formulations of NC interpretation that do not require a static, strict set of relation categories. One such alternative is *NC paraphrasing*, whose goal is to produce or recognize suitable paraphrases of NCs. For example, the paraphrase *part that makes up body* is a suitable paraphrase of *body part*, but *replacement part bought for body* is not (Shwartz and Dagan 2019). This task may be considered as a more faithful representation of NC interpretation than NC classification, as it leaves room to define novel and expressive types of NC relations as necessary to capture the nuance of any individual NC or group of such. However, this open-endedness cuts both ways: it is easy to imagine many generic paraphrases that, without being technically inaccurate descriptions of NCs, are neither especially informative nor helpful in distinguishing between NCs with very different relations. For example, *soup can*, *party member*, and *service price* could all be plausibly paraphrased as *N2 of N1*, even though the relationships between their constituents are quite distinct.

These NC paraphrasing concerns are at least partially addressed by the task defined by SemEval 2010 Task 9 (Butnariu et al. 2009) and extended by SemEval 2013 Task 4 (Hendrickx et al. 2013). These tasks define a popular approach to NC paraphrasing in which human annotators are asked to write paraphrases of NCs, and paraphrases are ranked in terms of how many annotators chose that paraphrase. In this case, some vague or vacuous NC paraphrases may still be created; but these will be accompanied by many other paraphrases, at least some of which are likely to be informative. NC paraphrasing systems may then be trained and evaluated on not just their abil-

[2] The first version of this taxonomy and associated dataset was released in Tratz and Hovy 2010, but my research uses the updated Tratz 2011 dataset, which added over 1k NCs and made minor revisions to the taxonomy.

ity to classify paraphrases as acceptable or not, but on their ability to rank these paraphrases (in terms of their likelihood to be chosen by annotators).

2.2 NC Interpretation Systems

2.2.1 Paraphrasing Approaches

A broad variety of NC interpretation systems have been developed in the last few decades. One class of systems, designed around the NC paraphrasing task, uses a small seed set of NCs, NC paraphrases, or both to learn patterns in how NCs are paraphrased, and find or assess more NC paraphrases. Nakov and Hearst 2006 used sets of NCs developed in other NC interpretation research to perform web searches with queries of the form "*N2* that[3] * *N1*" (where "*" is a wildcard symbol matching any word). They collected all instances of this pattern from each query's results, parsed each matching instance, and extracted the verb (and the preposition following it, if applicable) as a new valid paraphrase connecting N2 and N1.

Kim and Nakov 2011 followed a similar procedure, starting with either a small set of NCs or NC paraphrases as a seed. If starting with NC paraphrases, they began by issuing queries of the form "* that PATTERN *" (where PATTERN is an inflected verb) and parsed search results to find examples of nouns filling the wildcards ("*"). Alternatively, if starting with seed NCs, they issued queries of the form "*N2* that[4] * *N1*" (the same form as used by Nakov and Hearst) to extract NC paraphrases. This full process can be repeated indefinitely, extracting as many NCs and NC paraphrases as desired.[5]

[3] The "that" term in the query matches any one of "that," "which," "who."

[4] The "that" term in the query matches any one of "that," "which," "who," or the empty string.

[5] Kim and Nakov 2011 applied several filtering steps and heuristics to refine their collection of NCs and NC paraphrases between the NC-extraction and paraphrase-extraction phases; but for

Another interesting paraphrasing approach was developed by Nulty and Costello 2013 using the data from SemEval 2010 Task 9 (Butnariu et al. 2009), which contains almost 15k paraphrases of 355 distinct NCs. Their approach used a conditional probability model based on groups of paraphrases that are used for the same NC to score new candidate paraphrases. That is, for any NC, and any two paraphrases r_1, r_2 with probabilities $\Pr(r_1) > 0$ and $\Pr(r_2) > 0$ of being a paraphrase selected for the NC (i.e., each of r_1 and r_2 was provided as a paraphrase for the NC by at least one annotator in SemEval 2010 Task 9), the conditional probability that any given annotator would select r_1 as a paraphrase for the NC given that r_2 has been selected, $\Pr(r_1|r_2)$, may be computed using the definition of conditional probability:

$$\Pr(r_1|r_2) = \frac{\Pr(r_1 \cap r_2)}{\Pr(r_2)}$$

and the overall score for any such paraphrase r_1 is given by:

$$score(r_1) = \sum_{r_2 \in R} \Pr(r_1|r_2)$$

Indeed, $score(r_i)$ may be computed with respect any subset $\{r_i\}$ of R for use in ranking them, which makes it possible to test the probability model on the NC paraphrase ranking task defined in SemEval 2010 Task 9 (Butnariu et al. 2009).

Next, Nulty and Costello 2013 demonstrated that, starting with a small set S of seed paraphrases, this probability model could be used to score a candidate paraphrases r for an NC, for $r \in R$ where R is the list of *all* paraphrases, as follows:

brevity's sake, I am not elaborating them here.

$$score(r) = \sum_{s \in S} \Pr(r|s)$$

This list of all $r \in R$ can then be sorted by score, and the top scoring $r \in R$ is selected as a candidate paraphrase for the NC.

2.2.2 Feature-Selection Approaches

Feature-selection approaches focus on representing NCs with broad, diverse sets of handpicked features related to each NC, and using machine learning techniques to determine which features contribute to classification (and which do not). These features can be gathered from a variety of databases, dictionaries, thesauri, and other data collections or reference materials, combining features from as many sources as one thinks may be helpful in classification. One example of this approach was implemented by Tratz and Hovy 2010, who developed NC representations which included (among others) such diverse features as the set of all part-of-speech roles (other than noun) that each NC constituent can fill, synonyms of both NC constituents, the last 2-3 letters of each constituent, and even a binary $\{0, 1\}$ feature indicating whether one constituent is referenced in dictionary definitions of the other. Another NC feature-selection system was created by Nastase and Szpakowicz 2003, whose NC representations included features such as each NC constituent's part of speech, the lemma of each constituent (e.g., the lemma of *container* is *contain*), the part of speech of each constituent's lemma, and several other lexical and syntactic features related to NC constituents.

2.2.3 Kernel Methods

Ó Séaghdha and Copestake 2013 experimented with a variety of *kernel methods*, which they used to map representations of NCs into spaces which make various forms of similarities between NCs more clear, and trained SVMs to classify NCs in these spaces. In particular, they experimented with kernels that computed the *lexical similarity* and *relational similarity* of NCs. The lexical similarity of NCs are defined in terms of the similarity of the contexts in which NC constituents may be found (e.g., the NCs *history book* and *geography article* would be judged to be lexically similar so long as *history* and *geography* are both frequently seen around the same words, and if the same is true of *book* and *article*). Relational similarity, also known as *analogical similarity* (see Turney 2006), between two NCs is defined with respect to the similarity of the contexts in which *both* constituents of each NC appear (e.g., *wood floor* and *brick wall* would be recognized as relationally similar if the phrases and sentences in which both constituents appear often contain the same words). In experimenting with kernel methods based on these notions of similarity, Ó Séaghdha and Copestake found that systems which combined kernels constructed in terms of lexical similarity with those built around relational similarity almost always outperformed systems using only one of these types of kernels.

2.3 Contemporary Work

2.3.1 Compositional Approaches

Dima 2016 implements several versions of the *compositional approach* to NC classification, where a compositional function f is learned that transforms d-dimensional word embeddings $\vec{u}, \vec{v} \in \mathbb{R}^d$ (corresponding to some NC (n_1, n_2), and drawn from the set of 300-dimensional pre-trained GloVe embeddings; see Pennington, Socher, and Manning 2014) into a joint compositional vector, $\vec{p}$ (i.e., $f(\vec{u}, \vec{v}) = \vec{p}$). The compositional vector $\vec{p}$ is then passed to a classifier, which maps individual compositional vectors to relations. Dima experimented with a few different functions, including the function f defined by the *full-additive model* (developed by Zanzotto et al. 2010), for which square matrices $\mathbf{A}, \mathbf{B} \in R^{d \times d}$ are learned, where:

$$f(\vec{u}, \vec{v}) = \mathbf{A}\vec{u} + \mathbf{B}\vec{v} = \vec{p}$$

Another function f explored by Dima 2016 is the *matrix model* developed by Socher et al. 2011, where:

$$f(\vec{u}, \vec{v}) = g(\mathbf{W}[\vec{u}; \vec{v}] + \vec{b}) = \vec{p}$$

for pre-established non-linear function g (e.g., the element-wise hyperbolic tangent function, tanh), the concatenation of $(\vec{u}, \vec{v})$, $[\vec{u}; \vec{v}] \in \mathbb{R}^{2d}$, learned matrix $\mathbf{W} \in \mathbb{R}^{d \times 2d}$, and learned bias term $\vec{b} \in \mathbb{R}^{d}$.

2.3.2 Path-Based Approach

Shwartz and Waterson 2018 developed a *path-based approach* to NC classification, where NCs are represented by (syntactic) dependency paths connecting their constituents, which are extracted from a large text corpus. For example, for the NC *coffee cup*, dependency paths might include *cup of coffee*, *cup containing coffee*, *coffee in cup*, etc. These paths are represented by a concatenation of the following features for each word in the path: its pre-trained GloVe embedding, its part of speech, the dependency relation between it and its parent in the path, and the direction of that dependency relation (i.e., either *left* or *right*, depending on whether the word's parent comes before or after it in the text). Each path is then passed into an LSTM (Hochreiter and Schmidhuber 1997), which encodes it as a path embedding $\vec{p_i} \in \mathbb{R}^d$. For each NC, the 1000 most common paths (in the corpus) for that NC are computed, and then averaged (weighted by frequency) to yield the path-based representation for that NC. That is, for all path embeddings $\vec{p_i}$ of an NC, where the frequency of $\vec{p_i}$ is $f_{\vec{p_i}}$, the path-based representation of the NC is given by:

$$\vec{p} = \frac{\sum_{\vec{p_i}} (f_{\vec{p_i}} \cdot \vec{p_i})}{\sum_{\vec{p_i}} f_{\vec{p_i}}}$$

At the end of both Dima 2016's and Shwartz and Waterson 2018's approaches, the representation of a given NC, $\vec{p}$, is passed as the input to shallow feed-forward neural networks which are trained to map all such $\vec{p}$ to relations. On its own, the path-based model of Shwartz and Waterson 2018 does not achieve particularly impressive performance; but the *integrated* version of this model (where the pre-trained GloVe embeddings for each of the NCs constituents are simply passed as input to the neural network alongside the NC's path embedding, $\vec{p}$) scores much higher.

Each of these approaches is state-of-the-art for a different split of the Tratz 2011 dataset. If the dataset is split randomly into training, validation, and test sets, the compositional approach of Dima 2016 performs better. However, there is a phenomenon, referred to as *lexical memorization*, that concerned both Shwartz and Waterson 2018 and Levy et al. 2015, wherein a classifier *memorizes* the relation that is exhibited by the most NCs with a given head noun, and may achieve relatively high performance by simply reporting the most common relation among all NCs in the training data with the same head. For example, any NC with the head *juice* (e.g., *fruit juice*, *grape juice*, *orange juice*, etc.) is very likely to exhibit the *ingredient* relation, so any classifier which encounters even one *juice* NC while training, and can remember its relation, is essentially guaranteed to correctly classify all future *juice* NCs.

According to Shwartz and Waterson 2018, the paths in their path-based model may be understood as paraphrases of the meanings of NCs. However, these paraphrases are often relatively uninformative (e.g., *cup of coffee* or *coffee in cup* do not go much further toward elucidating the meaning of the NC *coffee cup* than the NC on its own). Further, the fact that averaged NC path embeddings are frequency-weighted means that the less frequent a path connecting NC components is, the smaller a role it plays in final NC path embeddings. As a result, longer and more specific paths – which, by virtue of their length, have the potential to contain a great deal of elaborating information – will not contribute as much to final path embeddings as shorter, potentially less informative paths. Definitional templates, on the other hand, are designed to explicitly elaborate a relation (i.e., its *definition*), and this information is not distorted by frequency weighting. Thus, the templating approach may offer some advantages over the path-based approach.

3 Methods

3.1 Overview

I created definitional templates to cover the range of coarse-grained relations in the Tratz 2011 dataset, and developed several classification models in which to apply them. I then tested these templates in the context of each model to determine the extent to which they contribute to NC relation classification. The processes for developing and testing these templates, as well as the templates themselves, are elaborated below.

3.2 Data

This research used the Tratz 2011 dataset, which labeled ∼19k NCs according to an NC relation taxonomy containing 11 coarse-grained relations and 37 fine-grained relation sub-categories (with all NCs having one coarse-grained and one fine-grained label). This dataset was broken into a train, evaluation, and validation split by Shwartz and Waterson 2018 using a 75:20:5 ratio (which they referred to as their *random* split), which is the split I used for all experiments. The *Cause* coarse-grained relation of Shwartz and Waterson's data, and its sole fine-grained sub-category, *Experiencer of Experience*, were not listed by Tratz; however, Tratz listed a very similar fine-grained sub-category, *Experiencer + Cognition/Mental Experience*, under the coarse-grained relation of *Ownership, Employment & Use*. As a result, the *Cause* and *Ownership, Employment & Use* relations were merged under *Ownership, Employment & Use*, preserving Shwartz and Waterson 2018's fine-grained sub-category label of *Experiencer of Experience*.

The Tratz 2011 dataset contained one coarse-grained relation, *Other* (containing 1982 NCs, about 10% of the dataset), for which – due to its catch-all nature – a template could not be defined. As such, I did not use it in this work. This prevents any one-to-one comparisons between the results discussed in §4, and those of Shwartz and Waterson 2018, which all included the *Other* relation.

3.3 Templates

I developed an initial set of definitional templates, T_0, by hand-crafting a template to encapsulate the definition of each coarse-grained NC relation type. If a single template appropriately captured the meaning of a coarse-grained relation with respect to NCs of all its fine-grained sub-relations, I did not create any additional templates for that relation. However, if a coarse-grained relation had fine-grained sub-categories with particularly distinct meanings, I created multiple templates to match as many fine-grained relations as necessary.

Each template in T_0 was tested in the LM probability estimator (see §3.4.1), and was only kept if it improved the overall performance of the classifier across all coarse-grained relations (by macro F1-score), or was the only template for that relation. That is, for the subset of templates in T_0 corresponding to coarse-grained relation r denoted as $T_{0,r}$, the final set of templates T was constructed as follows:

Algorithm 1 Template selection algorithm

```
Initialize $T \leftarrow \varnothing$
for $t \in T_0$ do
    Let $r$ be the coarse-grained relation defined by $t$
    Let $p$ be the LM probability estimator's performance using all templates in $T_0$
    Let $p'$ be the LM probability estimator's performance using all templates in $T_0 - \{t\}$
    if $p > p'$ or $|T_{0,r}| = 1$ then
        $T \leftarrow T \cup \{t\}$ [6]
    end if
end for
```

Thus, templates in T_0 which did not contribute to the LM probability estimator's performance are not included in T. This process was repeated as necessary to test new or modified templates, yielding the final set of templates, T, in Table 1:

[6] In cases where $|T_{0,r}| > 1$, but adding any template in $T_{0,r}$ to T would decrease performance, only the template in $T_{0,r}$ which decreased performance the least was added.

Coarse-Grained Relation	Template(s)	Example
Attribute	The N1 N2 is a N1 and a N2.	island country
Causal	The N1 caused the N2. The N1 performed the N2.	job stress police raid
Complement	The N1 N2 is the N2 of the N1.	homicide victim
Containment	The N2 is made of N1.	gold jewelry
Location & Whole+Part	The N2 is located in the N1.	household appliance
Objective	They will N2 the N1.[7]	carbon tax
Ownership, Employment & Use	The N2 works for the N1. The N1 experiences N2.	university professor investor enthusiasm
Purpose	The N2 is used for N1. The N2 supervises the N1.	construction vehicle highway official
Time	The N2 occurs during N1.	evening performance
Topical	The N2 is about the N1.	lease agreement

Table 1: Final template set, T

3.4 Models

I developed three types of classifiers that apply these definitional templates for NC classification: a *language model (LM) probability estimator*, a *feed-forward neural network*, and a *fine-tuned LM*. Feed-forward neural networks and fine-tuned LMs require hyperparameters for their learning/fine-tuning processes, and these hyperparameters were selected on the basis of the performance they yielded on the validation set. Training continued as long as model performance increased across epochs, as measured by weighted F1-scores on the validation set.

[7] For the *Objective* template, N2 was lemmatized (or, failing that, stemmed) as a verb, if possible; e.g., *flood prevention* fills the template as "They will *prevent* the *flood*."

3.4.1 LM Probability Estimator

The LM probability estimator classifies NCs by estimating the probability that their constituent nouns fill each individual template. They then classify the NC as having the relation defined by the template that the NC has the greatest probability of filling. The probability of an NC filling a given template is estimated by filling the N1 and N2 slots of the template with `[MASK]` tokens, using BERT (Devlin et al. 2018) to compute the masked language model (LM) loss[8] of the template with respect to the constituents of the NC. This loss is normalized with respect to the template's average loss, and dividing this normalized loss by the sum of all such losses for the NC across all templates.

Specifically, for each template t, $\bar{L_t}$ is defined as the ratio of the smallest masked LM loss on t among all NCs to the sum of the masked LM losses on t for all NCs; i.e.:

$$\bar{L_t} = min_{NC_i}\left\{\frac{loss(NC_i, t)}{\sum_{NC_j} loss(NC_j, t)}\right\}$$

The normalized template loss of each NC_i with respect to template t, $z_{NC_{i,t}}$, is defined as:

$$z_{NC_{i,t}} = \frac{\bar{L_t}}{loss(NC_i, t)}$$

The estimated probability that NC_i matches template t, denoted $\Pr(t|NC_i)$, is computed by dividing $z_{NC_{i,t}}$ by the sum of the normalized losses of NC_i across all templates, yielding:

[8] Masked LM loss is used exclusively as a scoring metric, and not for any form of training.

$$\Pr(t|NC_i) = \frac{z_{NC_{i,t}}}{\sum_{t' \in T} z_{NC_{i,t'}}}$$

Finally, each NC_i is classified as having the relation corresponding to the to template t which yields the greatest $\Pr(t|NC_i)$.

3.4.2 Feed-Forward Neural Network

I developed three variations of a feed-forward neural network classifier: an NC-only model, a template-only model, and a hybrid template/NC model. Following Shwartz and Waterson 2018, all variants have one hidden layer of width one-half the size of its input layer, use a linear activation function, are randomly initialized, and are trained using the Adam optimizer (Kingma and Ba 2017).

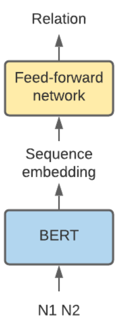

Figure 1: Feed-forward NC-only variant

The NC-only variant's inputs are the 768-dimensional sequence-level BERT (Devlin et al. 2018) embeddings (i.e., the `[CLS]` token) produced by feeding an NC into BERT (base, uncased), and has an output layer of width 10 (for the 10 relation classes). It is trained on the task of mapping these sequence embeddings directly to relation classes.

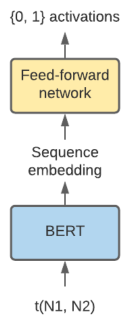

Figure 2: Feed-forward template-only variant

The template-only variant's inputs are the sequence-level BERT (Devlin et al. 2018) embeddings produced by feeding filled templates (denoted t(N1, N2) in Figure 2) into BERT (base, uncased). It has an output layer of width 1, and is trained on the binary classification task of determining whether a filled template expresses the relation of the NC it contains.[9] Classification is performed by obtaining the output layer activation corresponding to the positive label, 1 (i.e., the label for filled templates with a relation matching the NC they were filled with), for each template, and classifying the NC as having the relation which corresponds to the template with the highest positive-label activation.

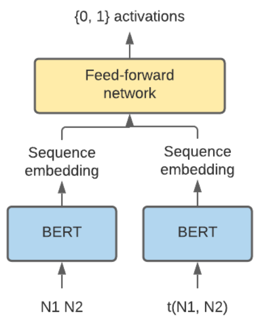

Figure 3: Feed-forward hybrid variant

[9] A single classifier is trained with respect to all templates, relations, and training NCs. Each training instance is labeled as 1 if it contains a template and NC associated with the same relation, and 0 otherwise, irrespective of the particular template or NC.

The hybrid variant's inputs are a concatenation of the inputs to both the NC-only and template-only variants (yielding a 1536-dimensional vector). It has an output layer of width 1, and its training and classification are performed in the same way as the template-only variant.

3.4.3 Fine-Tuned LM

As in the case of the feed-forward neural network classifier, I developed three variations of the fine-tuned LM classifier: an NC-only model, a template-only model, and a hybrid template/NC model. However, in the place of BERT embeddings (Devlin et al. 2018), these models instead take a string of word tokens as input. Each variant begins as the BERT-base (uncased; Devlin et al.) model, and is fine-tuned on a different task.

I fine-tuned BERT-base (uncased; Devlin et al. 2018) by removing its output layer, replacing it with a new, randomly initialized final layer (without modifying any other component of the network), and training the modified model on one of two fine-tuning tasks described below. This new final layer has an input dimensionality corresponding to the width of the final hidden layer of the pre-trained network – which, for BERT-base (uncased), is 768 – and an output dimensionality corresponding to the number of output classes required for the fine-tuning task – which, for these fine-tuned LMs, is either 10 (for the NC-only variant) or 1 (for the template-only and hybrid variant).

The NC-only variant has an output dimensionality of 10 because it is trained to map NC inputs directly to relation classes (of which there are 10). The template-only and hybrid variants, like their analogues among the feed-forward network models, are trained on the binary classification task of determining whether a filled template expresses the relation of the NC it contains. The template-only variant's input is a filled template, whereas the hybrid variant's input is an A/B sequence consisting of a filled template, a separator (`[SEP]`) token, and the NC. (For examples of A/B sequences as BERT inputs, see the next-sentence prediction, sentence-pair classification, or question answering tasks described by Devlin et al. 2018.)

4 Results

If the definitional templating approach contributes to NC relation classification, then the template-only or hybrid variants of the feed-forward neural network and fine-tuned LM classifiers should outperform the NC-only variants. Alternatively, if templates are not useful in NC relation classification, then the template-only and hybrid variants should perform similarly to or worse than the NC-only variants.

Results from the feed-forward neural network models and fine-tuned LMs are shown in Table 2, Table 3, and Table 4 below, with the best results for each metric (among those each type of classifier) in boldface. (Due to time and resource constraints, all experiments could only be run once. It is possible that random initialization had nontrivial effects on these results – particularly in the case of the feed-forward neural network models – that might have become apparent across multiple runs.)

Classifier	Variant	Precision	Recall	F1-score
Feed-Forward Neural Network	NC-only	0.687	0.660	0.668
	Template-only	0.717	0.644	0.672
	Hybrid	**0.724**	**0.686**	**0.699**
Fine-Tuned LM	NC-only	0.722	**0.724**	0.717
	Template-only	0.723	0.721	**0.721**
	Hybrid	**0.724**	0.718	0.718

Table 2: Macro-averaged results

Classifier	Variant	F1-score (Accuracy)
Feed-Forward Neural Network	NC-only	0.706
	Template-only	0.704
	Hybrid	**0.732**
Fine-Tuned LM	NC-only	0.734
	Template-only	**0.749**
	Hybrid	0.743

Table 3: Micro-averaged results

Classifier	Variant	Precision	Recall	F1-score
Feed-Forward Neural Network	NC-only	0.705	0.706	0.701
	Template-only	0.711	0.704	0.701
	Hybrid	**0.739**	**0.732**	**0.729**
Fine-Tuned LM	NC-only	0.757	0.734	0.740
	Template-only	0.752	**0.749**	**0.750**
	Hybrid	**0.762**	0.743	0.749

Table 4: Weighted-average results

The macro-averaged results (Table 2) take the average of each relation class' precision, recall, and F1-scores for each relation class (irrespective of the number of NCs with any given relation); the micro-averaged results (Table 3) simply compute performance across all NCs regardless of their relation; and the weighted-average results (Table 4) compute precision, recall, and F1-scores across NCs with each relation, and average them together by weighting the figure for each relation according to the number of NCs with that relation.

4.1 Feed-Forward Neural Network Results

Among the feed-forward neural network results we see that, while the template-only variant does not always outperform the NC-only variant, the hybrid variant consistently does. This suggests that the embeddings generated by feeding either NCs or filled templates into BERT (Devlin et al. 2018) contain complementary information. Thus, a feed-forward neural network classifier which is able to access information from both of them, as the hybrid variant does, is able to outperform the variants which have access to only one of them (despite the fact that the hybrid variant has an expanded search space, as its hidden layer has a higher dimensionality than that of the other variants).

4.2 Fine-Tuned LM Results

4.2.1 Relation Analysis

The template-only and hybrid variants outperformed the NC-only variant by 0.9-1.5% across micro-averaged and weighted-average results while showing similar performance (with an improvement of only 0.3%-0.4% over the NC-only variant) among macro-averaged results. For this to be the case, they must be outperforming the NC-only variant with respect to relatively more common NC relation classes, while achieving similar performance or under-performing on the less common relation classes.

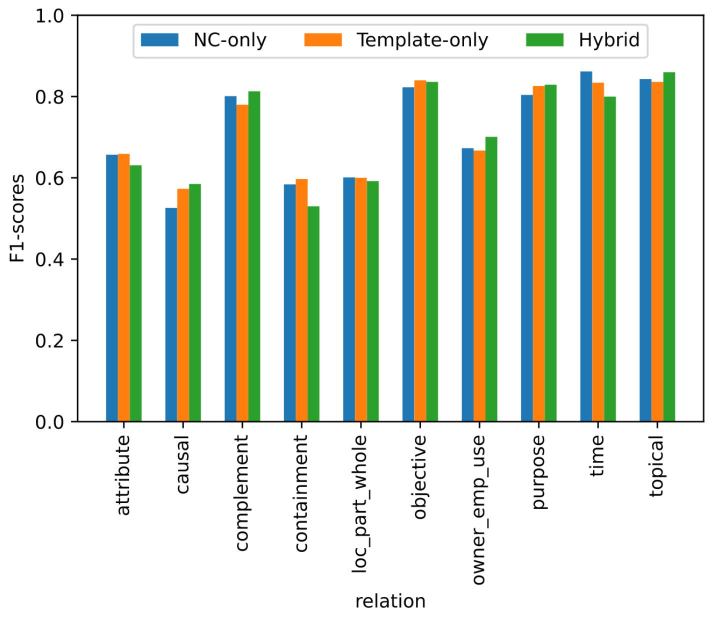

Figure 4: Fine-tuned LM F1-scores by relation

Consider the two largest relation classes, *Purpose* and *Objective*, which together represent 39% of the NCs in the Tratz 2011 dataset. As seen in Figure 4, both the template-only and hybrid variants have higher F1-scores than the NC-only variant on these relations. On the other hand, the NC-only variant has a higher F1-score than the other variants on the smallest relation class, *Time*, which makes up just 2.7% of the Tratz dataset. Thus, while each of the NC-only and template-only variants outperforms the other on exactly half of the NC relation classes, the template-only variant yields superior performance (measured by F1-score) on NCs representing 66% of the Tratz dataset (those with the *Attribute*, *Causal*, *Containment*, *Objective*, or *Purpose* relations), compared to 34% for the NC-only variant (including NCs with the *Complement*, *Location & Whole + Part*, *Ownership, Employment & Use*, *Time*, and *Topical* relations).

Interestingly, of the 3 relation classes with the highest overall fine-tuned LM performance (as measured by the average F1-score between all variants), *Topical*, *Objective*, and *Time*, the NC-only variant substantially outperforms the template-only variant on 2 of them (*Topical* and *Time*). Reciprocally, of the 3 relation classes with the *lowest* overall performance (*Causal*, *Containment*, and *Location & Whole + Part*), this pattern is reversed, with the template-only variant outperforming the NC-only variant on 2 of them (which also happen to be the lowest two, *Causal* and *Containment*). This pattern suggests that perhaps the switch from the NC-only to template-only fine-tuned LM brings down performance on the "easiest" relation classes (e.g., *Time*), while buttressing up performance on the most "difficult" (e.g., *Causal*). This idea is supported by the weighted-average results (Table 4), which show that the template-only variant has a substantially higher weighted-average recall than either of the other two variants, while also having the lowest weighted-average precision. Thus, it has broader overall coverage of the diverse range of NCs in the Tratz 2011 dataset than the NC-only (or hybrid) variants, but this coverage comes at the cost of higher precision.

4.2.2 Template-Only and Hybrid Variants

Unlike the feed-forward network variants, the template-only fine-tuned LM variant outperformed the hybrid variant, with both of these variants outperforming the NC-only variant (at least as measured by the micro-averaged results). One possible explanation for these results is that the fine-tuned LMs are better at attending to specific word-pairs in filled templates than the feed-forward networks. (After all, the fine-tuned LMs receive a tokenized word sequence as input, which will always contain the NC constituents, while the feed-forward networks simply receive one 768-dimensional embedding representing the entire word sequence – or two, in the case of the hybrid feed-forward network variant.) Should this be the case, it is conceivable that the template-only fine-tuned LM variant is able to attend to the NC constituents in a filled template, making the hybrid variant's inclusion of the lone NC as its second sequence input largely redundant.

Another possibility is that, while the template and NC inputs may still contain some complementary information (potentially leading to the difference in precision and recall between the two leading variants), some aspect of the experimental configuration or fine-tuning process might attenuate the marginal benefits of including them both. For example, it may be that BERT's (Devlin et al. 2018) A/B sequence input format is not particularly well-suited to this task, so while some presentation of both the filled template and the NC as inputs might have had a greater impact on the performance of a differently-configured hybrid variant, this impact was not apparent in the A/B sequence input configuration of the hybrid variant.

5 Conclusions

This research explored the use of definitional templates in NC relation classification, developing a variety of models to explore how they might be applied to this task and the extent to which they increase performance on it. Results from the feed-forward networks (see §4.1) indicated that including both templates and NCs as inputs to classifiers can produce complementary effects, leading to overall performance gains, relative to performance observed for classifiers with access to only one of these inputs. Results from the fine-tuned LMs (see §4.2) showed that introducing templates in the context of sophisticated deep learning architectures – such as that of BERT (Devlin et al. 2018) – can lead to more varied outcomes, but nonetheless improves performance on the whole.

5.1 Future Research

As discussed in §4.2.2, the hybrid fine-tuned LM did not see the same complementary effects from including both templates and NCs as was evident among the feed-forward network variant results. However, the possibility remains that these complementary effects might appear in the performance of fine-tuned LMs if alternative approaches are employed for including both templates and NCs as inputs to a fine-tuned LM. Testing such alternatives could help to determine whether feed-forward networks only exhibit complementary effects because they have a lesser capacity (relative to fine-tuned LMs) to attend to the presence of NCs in filled templates, or whether these effects may be extended to fine-tuned LMs as well.

One remaining question regarding the templating approach is how to handle NCs from the Tratz 2011 dataset with the *Other* relation. These NCs were not used in this research, because no *Other* template could be created; and as such, the results elaborated in §4 could not be directly compared to those of Shwartz and Waterson 2018 (as noted in §3.2). For direct comparison between leading NC relation classification approaches like those of Shwartz and Waterson to be possible, a method for handling NCs with the *Other* relation should be developed. For example, a threshold could be set such that, if no filled templates yielded a positive-label activation above the threshold, an NC would be classified as *Other*. This threshold could be empirically determined (so as to achieve the best balance between NCs of the *Other* threshold and all others), and could also be set at different values for each individual template for optimal performance.

This work could also be extended by developing a hybrid classifier incorporating other NC classification approaches (in particular, those of Shwartz and Waterson 2018 and Dima 2016) alongside the templating approach. Doing so might follow the blueprint of the hybrid feed-forward network variant (see §3.4.2) by concatenating multiple vector representations of NCs, and using this concatenated vector as input for a feed-forward neural network; but in this case, sequence-level embeddings of filled templates would be concatenated with Shwartz and Waterson's path embeddings and/or Dima's compositional vectors. Alternatively, an ensembling system could be constructed which contains one of the full template-based classifiers (perhaps that with the highest overall performance, the template-only fine-tuned LM), Shwartz and Waterson's path-based classifier, and any number of Dima's compositional classifiers, and classifies NCs according to a weighted sum of the votes from each of these. (The weights in this weighted sum should be learned, and might substitute $\{0, 1\}$ votes for each NC relation, with confidence scores from each classifier's output layer.)

Finally, additional templates could be developed for a variety of purposes. All the templates I developed corresponded to the coarse-grained relations of the Tratz 2011 dataset, but templates could also be developed for its fine-grained relations. Fine-grained template-based classifiers might even be able to contribute to coarse-grained classification by, for example, overriding a coarse-grained classifier in cases where a fine-grained classifier has a substantially higher positive-label activation for a fine-grained relation class than any of the coarse-grained positive-label activations on the part of the coarse-grained classifier. (Naturally, this would only change the outcome of classification in cases where the fine-grained classifier's highest confidence score corresponded to a fine-grained relation class under a different coarse-grained relation class than that which was indicated by the coarse-grained classifier; but I anticipate that such a system could be particularly helpful when handling NCs with some of the least frequent fine-grained categories.) Templates could also be created for other NC relation datasets (e.g., Girju et al. 2005, Ó Séaghdha 2008), or even applied to other tasks in lexical composition.[10]

[10] E.g., adjective attribute selection – see Hartung 2015's HeiPLAS dataset, and Shwartz and

6 References

Butnariu, Cristina, Su Nam Kim, Preslav Nakov, Diarmuid Ó Séaghdha, Stan Szpakowicz, and Tony Veale (June 2009). "SemEval-2010 Task 9: The Interpretation of Noun Compounds Using Paraphrasing Verbs and Prepositions". In: *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions (SEW-2009)*. Boulder, Colorado: Association for Computational Linguistics, pp. 100–105. URL: https://www.aclweb.org/anthology/W09-2416.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: arXiv: 1810.04805 [cs.CL].

Dima, Corina (Aug. 2016). "On the Compositionality and Semantic Interpretation of English Noun Compounds". In: *Proceedings of the 1st Workshop on Representation Learning for NLP*. Berlin, Germany: Association for Computational Linguistics, pp. 27–39. DOI: 10.18653/v1/W16-1604. URL: https://www.aclweb.org/anthology/W16-1604.

Downing, Pamela (1977). "On the creation and use of English compound nouns". In: *Language*, pp. 810–842.

Girju, Roxana, Dan Moldovan, Marta Tatu, and Daniel Antohe (Oct. 2005). "On the semantics of noun compounds". In: *Computer Speech & Language* 19, pp. 479–496. DOI: 10.1016/j.csl.2005.02.006.

Hartung, Matthias (Dec. 2015). "Distributional Semantic Models of Attribute Meaning in Adjectives and Nouns". PhD thesis. Institut für Computerlinguistik Ruprecht-Karls-Universität Heidelberg. URL: https://archiv.ub.uni-heidelberg.de/volltextserver/20013/.

Hendrickx, Iris, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Stan Szpakowicz, and Tony Veale (June 2013). "SemEval-2013 Task 4: Free Paraphrases

Dagan 2019's method for "[recasting] it as a binary classification task".

of Noun Compounds". In: *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*. Atlanta, Georgia, USA: Association for Computational Linguistics, pp. 138–143. URL: https://www.aclweb.org/anthology/S13-2025.

Hochreiter, Sepp and Jürgen Schmidhuber (1997). "Long short-term memory". In: *Neural computation* 9.8, pp. 1735–1780.

Kim, Su Nam and Preslav Nakov (July 2011). "Large-Scale Noun Compound Interpretation Using Bootstrapping and the Web as a Corpus". In: *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Edinburgh, Scotland, UK.: Association for Computational Linguistics, pp. 648–658. URL: https://www.aclweb.org/anthology/D11-1060.

Kingma, Diederik P. and Jimmy Ba (2017). *Adam: A Method for Stochastic Optimization*. arXiv: 1412.6980 [cs.LG].

Lees, R.B. (1960). *The Grammar of English Nominalizations*. Indiana University. Research Center in Anthropology, Folklore, and Linguistics. Publication 12. Indiana Univ. ISBN: 9783112048672. URL: https://books.google.com/books?id=dPsEAQAAIAAJ.

Levi, Judith N. (1978). *The Syntax and Semantics of Complex Nominals*. Academic Press New York. ISBN: 9780124451506.

Levy, Omer, Steffen Remus, Chris Biemann, and Ido Dagan (May 2015). "Do Supervised Distributional Methods Really Learn Lexical Inference Relations?" In: *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Denver, Colorado: Association for Computational Linguistics, pp. 970–976. DOI: 10.3115/v1/N15-1098. URL: https://www.aclweb.org/anthology/N15-1098.

Nakov, Preslav and Marti Hearst (2006). “Using verbs to characterize noun-noun relations”. In: *International conference on artificial intelligence: Methodology, systems, and applications*. Springer, pp. 233–244.

Nastase, Vivi and Stan Szpakowicz (2003). “Exploring noun-modifier semantic relations”. In: *Fifth international workshop on computational semantics (IWCS-5)*, pp. 285–301.

Nulty, Paul and Fintan Costello (July 2013). “General and specific paraphrases of semantic relations between nouns”. In: *Natural Language Engineering* 19. DOI: 10.1017/S1351324913000089.

Ó Séaghdha, Diarmuid (2008). *Learning compound noun semantics*. Tech. rep. University of Cambridge, Computer Laboratory.

Ó Séaghdha, Diarmuid and Ann A Copestake (2013). “Interpreting compound nouns with kernel methods.” In: *Natural Language Engineering* 19.3, pp. 331–356.

Pennington, Jeffrey, Richard Socher, and Christopher Manning (Oct. 2014). “GloVe: Global Vectors for Word Representation”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, pp. 1532–1543. DOI: 10.3115/v1/D14-1162. URL: https://www.aclweb.org/anthology/D14-1162.

Shwartz, Vered and Ido Dagan (Mar. 2019). “Still a Pain in the Neck: Evaluating Text Representations on Lexical Composition”. In: *Transactions of the Association for Computational Linguistics* 7, pp. 403–419. DOI: 10.1162/tacl_a_00277. URL: https://www.aclweb.org/anthology/Q19-1027.

Shwartz, Vered and Chris Waterson (June 2018). “Olive Oil is Made *of* Olives, Baby Oil is Made *for* Babies: Interpreting Noun Compounds Using Paraphrases in a Neural Model”. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. New Orleans, Louisiana: Association for

Computational Linguistics, pp. 218–224. DOI: 10.18653/v1/N18-2035. URL: https://www.aclweb.org/anthology/N18-2035.

Socher, Richard, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning (July 2011). “Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions”. In: *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Edinburgh, Scotland, UK.: Association for Computational Linguistics, pp. 151–161. URL: https://www.aclweb.org/anthology/D11-1014.

Tratz, Stephen (2011). “Semantically-enriched parsing for natural language understanding”. In: University of Southern California Digital Library (USC.DL). DOI: 10.25549/USCTHESES-C3-176191. URL: http://digitallibrary.usc.edu/cdm/ref/collection/p15799coll3/id/176191.

Tratz, Stephen and Eduard Hovy (July 2010). “A Taxonomy, Dataset, and Classifier for Automatic Noun Compound Interpretation”. In: *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Uppsala, Sweden: Association for Computational Linguistics, pp. 678–687. URL: https://www.aclweb.org/anthology/P10-1070.

Turney, Peter D (2006). “Similarity of semantic relations”. In: *Computational Linguistics* 32.3, pp. 379–416.

Warren, B. (1978). *Semantic Patterns of Noun-noun Compounds*. Acta Universitatis Gothoburgensis. Acta Universitatis Gothoburgensis. ISBN: 9789173460484. URL: https://books.google.com/books?id=glAIAQAAIAAJ.

Zanzotto, Fabio Massimo, Ioannis Korkontzelos, Francesca Fallucchi, and Suresh Manandhar (Aug. 2010). “Estimating Linear Models for Compositional Distributional Semantics”. In: *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*. Beijing, China: Coling 2010 Organizing Committee, pp. 1263–1271. URL: https://www.aclweb.org/anthology/C10-1142.

Name of Candidate:	Adam Davies
Birth date:	November 23, 1998
Birth place:	Normal, Illinois
Address:	1335 Hobble Creek Drive Springville, UT 84663