bComputational Semantics - LT2213

Seminar 1 – rule based computational semantics

Group 4 – Marie Vågsäter, Victoria Daniilidou, Eleni Fysikoudi and Anni Nieminen

**BRAINSTORMING / NOTES FROM THE ARTICLE:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Points I would like further clarified** | **Points that are interesting and/or relevant** | **Answers to teacher’s questions\*** | **Other stuff I would like to include in the report?** |
| **Marie** | **-**Lambda  -Better explanation regarding cooper storage?  -Clarifiaction regarding the use of skolem constant  (example: c1) | **-**practical application (?) by using python (NLTK), | **-**Challenges (translating NL to Logic):  Hard to capture what is actually meant (determining properties are then often reduced to symbolic manipulation)  -Confined to representing sentences with letters such as (P/Q) and do not focus on the internal structure (this is why first-order logic can be useful since it is expressive enough for good representation by examining models of first order formulas).  -Aspects(not captured by logic):  Context-dependent meaning in natural languages are often not captured with logic.  -Aspects (captured):  Sentence consitency, formally capturing truth conditions by using logical representations f a sentence(s).  -Underspecification:  -Cooper storage is a method used to manage the ambiguity of quantifier scope by delaying the decision on how to interpret quantifiers until more context is available.  - Other forms of underspecification, like lexical ambiguity, are handled through techniques that allow for multiple interpretations to be represented simultaneously  -Lambda  Lambda calculus is essential for representing meanings in a formal system, enabling the manipulation of functions and arguments within natural language processing.  -Theorem provers & model builders: Model builders are about finding a model where statements hold true, while theorem provers focus on proving logical relationships and consistencies between statements  -Reasoning  Humans reason logically although our inferences might not always align to theorem provers as human reasoning can be influenced by emotion and cognitive biases.  -NLP applications:  SQL tasks requiring precise semantic analysis, like question answering, information extraction, and automated reasoning systems, where understanding the logical structure of language is crucial. |  |
| **Victoria** | Gerald  (walk(x) & chew\_gum(x))[gerald/x] |  | Possible challenge: verbs like walk that can be sometimes transitive e.g John walks, John walks his dog.  -treated well: quantification  -not treated well : ambiguity    -underspecification in Cooper storage, we have a core semantic representation (main predicate) and the binding operators. We start from one binding and then when we get more context we continue with the next one. Example of quantifier scope underspecification:  Every kid has a Teddybear.  core: have(x,y)  Bindings:  λ.x.kid(x)  λ.y.teddybear(y)  -lambda calculus  Captures the compositional semantics, interaction between nps and predicates. A dog barks. \P.exists x.(dog(x) & P(x))  - model building tries to create a new model, given some set of sentences. If it succeeds, then we know that the set is consistent, since we have an existence proof of the model.  Counterexamples: when a model builder fails to find a model means that we have an inconsistent set. So the conclusions are not universal (i am not sure for that)  The theorem prover treats existing sentences in the thread as assumptions and attempts to prove φ; it is informative if no such proof can be found.  humans reason logically, but maybe due to some factors like metaphor they dont always follow thw formal rules of model builders |  |
| **Eleni** | -λ function quantified NP example 4.2 | -Mace4 finding counterexamples  -Prover9 finding proof of truth conditions | -ambiguity is hard to resolve with logic as well as scope resolution. (everybody admires someone),semantic meaning(humor hard to translate).  -treated well: inference,compositionality  -cooper storage deals with underspecification with a form of discourse analysis.(S-Retrieval, checking all binding operators) same with ambiguity  -λ calculus provides us with an invaluable tool for combining expressions of first-order logic,it’s very flexible and has a lot of power, composes meaning(compositionality.  -humans use logic but they are also highly ifluenced by context. They also guess sometimes and they are very flexible and creative.  NLP applications: question-answer systems, information extracting. Lambda calculus is used in coding as well for example when sorting( is this relevant??) |  |
| **Anni** | -sem values and why do we need them -  S[SEM=<?vp(?np)>] -> NP[SEM=?np] VP[SEM=?vp], compared to normal CFG rule expansion notations (that we learnt in formal linguistics)  -3.8 model building example – how exactly does the Mace4 find a scenario where the premises are true and the conclusion is false??  -3.8 skolem constant ?  -What benefits are there to using lambda as a binding operator compared to just using ∀ and ∃? | -First-order logic divides propositions into predicates and arguments/entities. Predicates take different number of arguments.  -Model checking: defining truth or falsity in a formula in a model.  -Model building (Mace) ensures that sets are consistent, can be used alongside the theorem prover (Prover9).  -When working with first-order logic, after valuation, we need to assign those ‘objects’ to variables in order to proceed to evaluating sentences’ truth values. | -Computers are good at determining consistency, thanks to logical operators. This is what humans also use to guide their judgements.  -Two of the challenges of denotation is the lack of context, i.e. with personal pronouns, and ambigue sentences.  -Model building requires detail as [*There is a woman that every man loves*, *Adam is a man*, *Eve is a woman*] ⇒ *Adam loves Eve* seems very intuitive to us humans but not to Mace4 model builders. -Inferencing with models can be seen as a bit black/white compared to how humans may interpret things(?) |  |

**\*** *What are challenges of translating natural language to logic (in general)?*

*Logics are formal languages each with specific properties: what aspects of natural language semantics are treated well with logic and what aspects are not captured?*

*How is underspecification of quantifier scope implemented in Cooper storage?*

*How about other forms of underspecification in natural language, e.g. lexical ambiguity?*

*Why do we need lambda calculus?*

*How can we use model builders and theorem provers (computational tools) to check validity of arguments?*

*Do humans also reason logically - do they make the same inferences as theorem provers and model builders?*

*In what NLP applications we would use this approach?*

**THE REPORT / THINGS DISCUSSED:**

Report made by Marie:  
Our group discussion started off by answering the questions provided. We then followed it with a discussion about the concepts we’ve had a litte bit of understanding and addressed this by sharing what we each thought about.

From our discussion we concluded that cooper storage is a method for managing the ambiguity of quantifier scope by postponing the binding of quantifiers until sufficient contextual information becomes available. This method addresses underspecification in natural language by maintaining a flexible framework for interpretation. Since it uses discourse referents and keeps alternative representations for each quantifier and then it narrows it down as the context gets more detailed. We figured that this also happens for lexical ambiguity.

We also discussed how theorem provers and model builders are tools used for proving logical relationships and consistency between statements. We differentiated it by concluding that model builders are used to construct models that satisfy a given set of logical statements, while theorem provers are designed to validate the truth of logical statements based on a set of axioms and inference rules.

We then talked about human reasoning versus computational logic. We argued that humans reason logically, but their inferences can be influenced by emotions and cognitive biases, which do not always align with theorem provers or model builders.

We concluded it by addressing some of the challenges we have encountered with the reading. Such as, propositional logic is not powerful enough to represent the meaning of natural language, even predicate logic has its limits, that’s why we need to combine it with lambda calculus. = Essential for representing meanings within a formal system, allowing for the manipulation of functions and arguments, thereby enhancing the expressiveness of natural language processing (NLP) and for compositionality.

We had a little bit of trouble understanding the skolem constant. We assumed that the skolem constant is created by the model to illustrate that there exists more than one woman from the example:

o y. all x. (woman(x) -> (x = y)) was tricky (a7) for us.

We talked about the challenges of translating natural language into logic, especially dealing with transitive/intransitive verbs (The bird sings, ), quantifier scope ambiguity (the sentence: everyone loves someone ) , and context-dependent meanings. (For example, keeping track of pronoun references), and metaphors etc.

We are interested in learning more about practical applications of logic as now this is very abstract. (Also, the reason why we even want machines to be able to make inferences the exact way as humans do?)

What benefits are there to using lambda as a binding operator compared to just using ∀ and ∃?

We would like to know more about the Applications in NLP:

Tasks that require precise semantic analysis, such as question answering, information extraction, and automated reasoning systems, benefit from understanding the logical structure of language.

Lambda Calculus in Coding: Besides theoretical applications, lambda calculus has practical applications in coding, such as sorting algorithms.

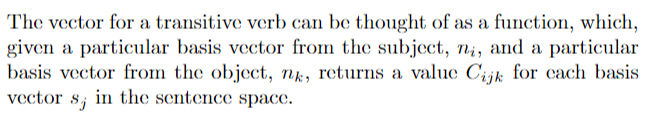
Computational Semantics - LT2213

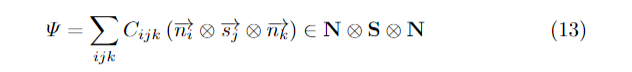
Seminar 2 – distributional representations

Group 4 – Marie Vågsäter, Victoria Daniilidou, Eleni Fysikoudi and Anni Nieminen

**BRAINSTORMING / NOTES FROM THE ARTICLE:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Points I would like further clarified** | **Points that are interesting and/or relevant** | **Answers to teacher’s questions / points to consider\*** | **Other stuff I would like to include in the report?** |
| **Marie** | Matrix multiplication of the noun vector  -more examples?  Examples of pregroup derivation with semantic types | Centroid vector | Notion of Meaning: Distributional representations in vector space models capture the meaning of words based on the contexts in which they occur. The key idea is that words with similar distributions in text are likely to have similar meanings.  Semantic Relations: These models capture various semantic relations such as synonymy, antonymy, hypernymy, and co-occurrence patterns between words.  Relation to Intuitive Semantic Relations: Distributional representations often align with our intuitive understanding of semantic relations in natural language. For example, words like "cat" and "dog" are expected to be closer in meaning than "cat" and "car" due to their similar contexts.  Relations Not Captured:  While distributional representations excel at capturing certain types of semantic relations based on context, they may struggle with capturing more abstract or knowledge-based relations that require deeper understanding.  Examples: Distributional models can effectively capture relationships like "king" is to "queen" as "man" is to "woman" based on co-occurrence patterns. However, they may struggle with capturing complex relational concepts like "cause and effect" or "temporal relationships."  Difference from Model-Theoretic Semantics: Distributional representations focus on statistical patterns in language use to derive meaning, while model-theoretic semantics in formal logic is based on truth conditions and logical entailment.  Sense and Reference:  In distributional models, words are represented by vectors in a high-dimensional space, capturing their sense based on context, rather than a fixed reference to real-world entities.    Benefits:  Robustness in capturing semantic similarities.  Ability to handle large-scale language data.  Flexibility in capturing polysemy and context-dependent meanings.  Challenges:  Difficulty in capturing abstract or knowledge-based relations.  Interpretability of high-dimensional vector spaces.  Handling data sparsity and noise.  Limitations and (Dangers):  Over-reliance on surface-level statistical patterns.  Lack of deep semantic understanding.  Vulnerability to biases present in training data.  Computational Resources and Tasks:    Resources:  Large text corpora, word embeddings (e.g., Word2Vec, GloVe), dimensionality reduction techniques, similarity measures.  Methods:  Machine learning algorithms, neural networks, dimensionality reduction techniques, clustering algorithms.  Tasks:  Word similarity, document clustering, sentiment analysis, information retrieval, machine translation.  Limitations: Distributional representations may not be suitable for tasks requiring logical reasoning, formal inference, or deep understanding of abstract concepts.    Benefits of Combining Representations:  Enhanced semantic expressiveness.  Ability to capture complex meanings of phrases and sentences.  Integration of syntactic and semantic information.  Challenges of Hybrid Models:  Ensuring compatibility between different types of representations.  Handling ambiguity and variability in language.  Interpretation of Distributional Representations: While distributional representations provide a geometric view of word meanings, interpreting them in terms of logical operations or formal semantics can be challenging.  Mapping between Representations: The success of mapping between distributional and formal representations depends on the complexity of the semantic relationships being captured and the expressiveness of the models used.  Effect of Grammar Variations: Different formal grammars may lead to variations in how semantic compositions are structured, affecting the mapping between distributional and formal representations |  |
| **Victoria** |  |  | Notion of meaning:  Words that occur in similar contexts tend to have similar meanings.  Meaning of words  --> vectors.  Semantic relations:  Synonyms, hyponums (eg. firm, subsidiary )  Distributional methods = the set of contexts, key for meaning representation.  Problems with ambiguity - the different senses of an ambiguous word could be  revealed by looking at the different  contexts in which the word occurs  Problem with synonyms: example of Curran experiment.  No modelled well: polysemous words  e.g the word goal  Maybe some dialectic phrases(idioms) ??  model-theoretic semantics, the semantic  vector for the verb can be thought of as encoding all the ways in which the verb  could interact with a subject and object, in order to produce a sentence  the distributional representations  used in cognitive science tend to follow the neural network model, in  which the basis vectors are effectively induced automatically from the data,  rather than specified in advance in terms of linguistic contextual elements  Benefit : efficient with large corpus  Challenge with the distributional representations is data sparsity: since  the basis vectors are so detailed, the counts for many combinations of target  word and basis vector may be unreliable (or zero), even for very large corpora.  Limitations: ??  Tools for representations:  use of SVD(singular value decomposition)  Latent Semantic Analysis  (LSA)  Use: words similarity  Compositionality:  sentence meaning is mediated by syntactic structure    Benefit : language processing applications  would bene t from a framework in which the meanings of whole phrases and  sentences can be easily compared,  benefit to the search engine  formal grammar and affect the mapping -  Syntax complex sentences / deal with ambiguity |  |
| **Eleni** | -Info about what and how BM25 works  -The windows method?? 3.1  **-**Are there alternatives to using distributional semantics? Isn’t that the field that NLP tasks and neural networks depends on??  - Has this formal representations and distributional ones been explored more since 2015? | **-** Distributional semantics represent well relations between words by capturing their closeness in meaning such as synonymy, antonymy,contextual similarity. For example, in an embedding space synonymous words are placed closed to each other and antonyms are placed farther.  Information retrieval works pretty well.  Word sense disambiguation is really useful , SVD, LSA , cosine similarity , tf idf,statistics, weigthting importance of words. | Distributional Semantics don't cover polysemy?? Do multiple meanings get lost and the most common is picked?  -Pragmatic relations e.g it's hot in here, phrasal verbs? Are they covered well?  -And they can perpetuate stereotypes which is an important problem.  Benefits : efficiency(reusability), dimensionality reduction(less computation)  Challenges : polysemy, oov, context dependence models are trained in specific contexts and then the same data are used in different contexts making it hard to perform well, computational cost  Risks : losing some meaning because vectors are dense and language is fine grained, bias amplification  Distributional semantics : don't focus on reference but in patterns of occurrence using maths, sense understood implicitly through the vectors, multiple senses  Logic : refers to the relationship between the real-word entity and the linguistic representation, there is room for only one sense to be referred to one linguistic expression.  Useful : text classification, sentiment analysis, nlu  Not as useful: syntactic parsing  Benefits : models which include syntactic components are proven to perform better, disambiguation  Challenges : it's very complex has it ever been done successfully for multiple sentences?  It affects how they interact with the distributional representations |  |
| **Anni** | -Normalizing the document length formula on page 7 (can we recap that). I guess we just divide by the length?  -SVD/LSA (statistical noice reduction), how it actually works, need recap from Machine Learning  -Schütze’s (1980) word meaning disambiguation clustering method (p.21) a bit had to understand. How will the centroid vector actually disambiguate the basis vectors?  -Again: would like to know more about the real-life applications  -Tensor product = matrix multiplication / tensor stacking?  Baroni & Zamparelli’s (2010) approaches (p. 26)  **Combinatory category grammar, CCG:**  -examples 9 and 10 p. 27  -cancelling?  -monoid?  -figure 9 p. 29 can we go through it together?  -the monoidal operator (above)  -tensor product space: when we multiply dimensionalities??  -dirac notation (p.34)  -Is the main idea behind CCG that if we wish to represent sentences in vector spaces -> instead of simply concatenating all the term vectors we need to consider the syntactic “blocks” of a sentence such as N+ADV, N+TRV+N etc. This way we can capture the meanings of sentences better? | -Vectors are a handy way of representing and computing similary / dissimilarity between words and their meaning.  -Because distance is such a relevant concept in semantics, it is useful to incorporate algebra and geometry into this subfield of linguistics.  -Distributional semantics main idea: words that occur together tend to have a similar meaning. Distributional models (in other words, looking into the context in which they occur) are a way of disambiguating word meanings.  -Vector space  -Problems in document retrieval: the documents are considered as bag-of-words, word order is relevant and should be taken into account.  -Document similarity: documents are similar if they contain many of the same words – term similarity – terms are similar if they occur in many of the same documents.  -When we narrow down the contextual window, we can no longer trust that similar words will co-occur (i.e. synonyms). Syntagmatig words words occur in the same text region (context window?) and paradigmatic share many of the same surrouding words. We can choose one of these approaches when we calculate term-term frequencies.  -By refining the building of the basis vectors (by including more linguistic information like the grammatical relations), we can also get more information of the meaning of the target word. However, this leads to sparse vectors.  -Something we need to consider when weighing our term-term vectors is that different words have different levels of importance (we shouldn’t blindly categorize certain words as stopwords without a focus on our target word) | -Distributional representations look at the context and use that to determine the semantic representation of meaning.  -Calculating similarity and dissimilarity is very easy. Distributional models capture both attributional similarity (maybe better?) as well as relational similarity. Analogies.  -Possible problems with capturing meaning: words with several meanings, sarcasm? Capturing meaning in sentences, documents?  -Vector space models are used for: document retrieval, based on measuring simply the similarity between query vector and document vector.  -These models can be used to study synonymity and mine texts, but the performances (at least in Curran (2004)) are not perfect as they mix up synonyms with antonyms, hyponyms etc.  However, even this level of performance can improve the user satisfaction for instance in search queries (expanding the search results). -If we can expand to represent whole sentences as vectors, we are able to produce even more precise search results. | It seems like the options of how the semantic relations and the way terms / documents can be represented by distributional models is very vast. This is why we need to have a clear idea beforehand of the precise relations we want to capture before we fine-tune our models / decide how to represent our data. |





\* **-What notion of meaning is represented by distributional representations?**

Distributional representations look at the context and use that to determine the semantic representation of meaning.

**-What semantic relations do they capture?**

Synonyms, hyponyms, antonyms, hypernyms- depends on the distance inside the vector

**-How do these relate to the semantic relations we intuitively recognise in natural language?**

Distributional representations often align with our intuitive understanding of semantic relations in natural language. For example, words like "cat" and "dog" are expected to be closer in meaning than "cat" and "car" due to their similar contexts. However, humans see words more holistically through social and cultural factors.

-Are there relations that they do not capture? Polysemous words (e.g goal ), metaphors, idioms, phrasal verbs.

-Think of examples in natural language that can modelled well with distributional relations and examples that cannot be. (previous 2)

**-How does this notion of meaning different from that taken in model-theoretic semantics that we looked at earlier?**

**-Sense and Reference:**

Distributional semantics : don't focus on reference but in patterns of occurrence using maths, sense understood implicitly through the vectors, multiple senses

Logic : refers to the relationship between the real-word entity and the linguistic representation, there is room for only one sense to be referred to one linguistic expression.

In distributional models, words are represented by vectors in a high-dimensional space, capturing their sense based on context, rather than a fixed reference to real-world entities.

**-What are the main ... for representing meaning of natural language this way?**

**-benefits ->** Easy and fast to compute simple semantic tasks. Useful applications in real-life such as document retrieval and comparison, web-queries etc.

**-challenges ->** data sparsity: since the basis vectors are so detailed, the counts for many combinations of target word and basis vector may be unreliable (or zero), even for very large corpora.

**limitations (and dangers!) ->** losing some meaning because vectors are dense and language is fine grained and detailed, bias amplification (there have been studies that have shown that stereotypes are being perpetuated e.g. women are less often associated with the word job, career and men less with family, house)

**-What computational resources, tools and methods do we use to create these representations?**

use of SVD(singular value decomposition) ,Latent Semantic Analysis (LSA) (check again), weighing the importance of words with tf(term frequency) and idf (inverse document frequency), cosine similarity

**-For what tasks can we use these representations? For what tasks cannot we use them?**

-Document retrieval and comparison, data mining, classification tasks and sentiment analysis, we can also study diachronic changes in the semantics of words

-Maybe we can’t use distributional representations for syntactic parsing

**-What would be alternative representations?**

Are there alternatives to using distributional semantics? Isn’t that the field that NLP tasks and neural networks depends on?? Are there different representations that are used nowadays?

**-What are the reasons and benefits of combining formal representations with distributional ones?**

We can take into account the sentence structure and how the words are combined (compositionality) so that we consider and study larger linguistic units and their meaning instead of just concatenated word representations.

**-What do you think are the biggest challenges of such hybrid models and representations?**

- very complex models that require a lot of computational cost

- how to make sure the vectors don’t become too sparse when adding this information

**-To what degree can we interpret distributional representations?**

We can make interpretations to some extent, but we also need to acknowledge the limitations discussed above (data sparsity, biases, characteristics of natural language challenging for machines).

**-How does this relate to how well a mapping between two types of representations can be achieved?**

The success of mapping between distributional and formal representations depends on the complexity of the semantic relationships being captured and the expressiveness of the models used.

**-There are several different ways to write a formal grammar. How would this affect the mapping?**

Different formal grammars may lead to variations in how semantic compositions are structured, affecting the mapping between distributional and formal representation.

DISCUSS :

DIFFERENCE BETWEEN COMPOSITIONALITY AND DISTRIBUTIONAL SEMANTICS

How will the centroid vector actually disambiguate the basis vectors?

SVD/LSA??

Has anyone successfully ever made a model that combines compositionality and distributional representations?

Computational Semantics - LT2213

Seminar 3 – distributed representations

Group 4 – Marie Vågsäter, Victoria Daniilidou, Eleni Fysikoudi and Anni Nieminen

**BRAINSTORMING / NOTES FROM THE ARTICLE:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Points I would like further clarified** | **Points that are interesting and/or relevant** | **Answers to teacher’s questions / points to consider\*** | **Other stuff I would like to include in the report?** |
| **Marie** |  | **For the 1st paper:**  -the importance of distributed word representations in capturing semantic relationships between words and enhancing the overall language modeling process.  **For the 2nd paper:**  **-**I was interested in how **t**he study conducted experiments to evaluate grounded language models' success when certain word compositions are excluded from training. | **First paper:**  Concepts: One-hot encoding:  method used to represent categorical data, (words in natural language processing.) Each word is represented as a sparse vector where all elements are zero except for one element corresponding to the word's index in the vocabulary, which is set to one.  Dense embeddings:  low-dimensional, continuous representations of words that capture semantic relationships and contextual information.  Distributional word vectors/matrices: These represent words based on their distributional properties in a corpus. Each word is represented as a vector or matrix capturing its co-occurrence statistics with other words in the corpus.  Word vectors with dimensionality reduction: These involve techniques like Latent Semantic Analysis (LSA) or Latent Dirichlet Allocation (LDA) to reduce the dimensionality of word vectors while preserving semantic information.  Word embeddings: Word embeddings are dense, low-dimensional representations of words learned from large corpora using techniques like Word2Vec, GloVe, or FastText. They capture semantic relationships between words and are often used in neural network models for various NLP tasks.  Benefits:  Word embeddings capture semantic relationships and similarities between words.  Weakness: Word embeddings may struggle with capturing rare or out-of-vocabulary words.  Word2Vec:   includes two models - (CBOW) and Skip-gram, which learn word embeddings by predicting context words given a target word or vice versa  GloVe :  a model that learns word embeddings by factorizing the word co-occurrence matrix to capture global word-word relationships.  LSTM:  are recurrent neural networks that can learn word embeddings as part of a larger language model.  Probabilistic language model:  aims to estimate the probability of a sequence of words in a language.  Relation to word embeddings:  Word embeddings play a crucial role in probabilistic language models by providing dense, continuous representations of words that capture semantic relationships. These embeddings help the model understand the context and meaning of words in a given sequence, improving the model's ability to predict the next word or generate coherent text.  **Second Paper:**  1. Compositionality can be learned from data using neural networks in language models. It is demonstrated that by training on extensive language datasets, neural networks can effectively capture compositional structures.  **-**done by conditioning language models on perceptual features and training them on sequences of words referring to spatial scenes, the models can learn to generate and interpret novel sentences based on the meanings of their individual parts.  **2.** Grounding a language model in perceptual information or locations is crucial for capturing the semantics of composed phrases and individual words. By conditioning the model on perceptual representations, it gains a better understanding of how each part of a phrase relates to the corresponding perceptual features. This grounding enables the model to produce precise descriptions of spatial scenes and improves its capacity to handle novel composed and decomposed descriptions.  3.  Distributional composed representations, formed by combining word embeddings with multimodal embeddings, are intended to capture the connections between words and perceptual features. These representations strive to render the meanings of composed phrases interpretable by leveraging the underlying perceptual information. Through training neural networks on these representations, researchers can assess the model's proficiency in comprehending and producing meaningful descriptions of spatial scenes.  4.  The representations learned from data may not always align perfectly with traditional compositional rules in formal semantics. While neural networks can capture compositional structures and learn to generate meaningful sequences based on training data, the exact nature of these representations may differ from what is expected based on formal compositional rules  5.  I think I agree because neural networks can be considered compositional models because they are composed of neurons that work together to define representations of input data. Each unit in a neural network contributes to the overall representation of the input, and the network as a whole can learn complex compositional relationships between different features.  6.  Neural networks have the capability to learn compositional functions to a certain extent, but their alignment with formal semantics can vary. These networks are proficient at capturing patterns and relationships in data by training on extensive datasets, which allows them to understand compositional structures in language and other domains. However, the learned functions may not always precisely adhere to the rules and principles of formal semantics. |  |
| **Viktoria** | **Different use of dense and sparse embeddings?**  **Difference between compositionality(words and syntax) and grounded compositionality (spatial relationship between the words)** | **What are the challenges in grounding language model in perceptual information?**  **How can we enhance the interpretability of the distributional representations??** | **Syntagmatic relations** –the linear sequence of the words give the meaning / syntax  **Paradigmatic relations** - the set of elements that can take the same semantic position within a sentence  **one-hot vector representation** - in nlp, a one-hot vector is a 1 × N matrix (vector) used to distinguish each word in a vocabulary from every other word in the vocabulary. The vector consists of 0s in all cells with the exception of a single 1 in a cell used uniquely to identify the word.  Dense vectors are embeddings from neural networks that, when combined in an ordered array, capture the semantics of the input text  Word embeddings can be generated using unsupervised learning algorithms such as Word2vec, GloVe and Continuous Bag of Words (CBOW) is a popular natural language processing technique used to generate word embeddings / predicts a target word given its surrounding context words.  Distributional word vectors – words with similar distribution in a text have also similar meaning    Word vectors dimensionality reduction – SVD ?? previous seminar  Word embeddings are a way of representing words as vectors in a multi-dimensional space, where the distance and direction between vectors reflect the similarity and relationships among the corresponding words.  Benefits of word embeddings : Capturing Semantics and solving the curse of dimensionality / less sparse , difficulties with words that dont exist in the training and dont have embeddings (PROBLEM ASSIGNMENT 2?)  kind of word embeddings ???  Like gloVe and Word2Vec(skipgram model / CBoW)??  Probabilistic language model - estimate a probability distribution of sequences of words based on observable samples from language production, by estimating conditional probabilities of words with a categorical distribution. These models learn to predict the probability of a word given its context so we need the word embeddings for that.  Compositionality / additive compositionality (mikolov example ), if “Volga River” appears frequently in the same sentence together with the words “Russian” and “river”, the sum of these two word vectors will result in such a feature vector that is close to the vector of “Volga River”.  Grounding language models in perceptual information like images in order to capture grounded compositionality  their model is capable of grounding novel compositions and also predicting grounding of single words while only learning from compositions.  distributional composed representations interpretable?? This depends on the process of creating those representations and the tools we use (dense embeddings/ dimensionality )  Statistical learning doesnt give the compositionality of the words but the patterns that are followed in the cooccurrence of words and phrases  Neural networks - compositional models , hidden layers extract features |  |
| **Eleni** | Continuous variables and discrete what are these two categories?  The mathematical points of creating neural networks are quite a challenge  What kind of embeddings does GPT use? it seems too advance to be using GloVe or Word2Vec? What about other large models?  Are there other word embeddings more advanced? | The curse of dimensionality and the way it is solved. Is it still a problem today with neural networks? | -Distributional vectors: made on distributional hypothesis mostly with PMI(pointwise mutual information) and they are high-dimensional and sparse.  Benefits : ?  Drawbacks : very sparse , and many dimensions -> more expensive, less practical  -Dimensionality reduction vectors: similar to distributional but have undergone SVD technique and are more dense while keeping the most semantically important info.  Benefits: can be efficient  Drawbacks: loss of meaning, hard to handle oov  -Word embeddings: dense and low-dimensional made with neural networks from large text data. E.g GloVe, Word2vec  Benefits: capture well semantic information, generalize, good performance  Drawbacks : computationally more expensive and complex, oov words, data dependency(bias)  -A probabilistic language model is a statistical model that assigns probabilities to a sequence of words. Embeddings are often used in neural networks as inputs and outputs.  -Compositionality can be learned through gradient learning from data if the data are informative enough.  -Grounding a language model can help enhance contextual understanding and the meaning of words in general. It can be used to achieve compositionality.  -Distributionally composed representations aren’t the easiest to interpret as they are sometimes created with very complex models, dimensionality might also make it harder.  -Statistical learning is very different as there are no linguistic rules but mathematical dependencies but maybe they are similar in that they both try to capture distributional rules.  -Neural networks depend on data to find compositional functions and even though they do in an extent, they are more general and harder to interpret.  - Neural networks are compositional models in the sense of they have different layers, inputs and outputs and try to capture compositionality in a way. | The bag-of-words technique doesn’t seem to incorporate a lot of compositionality as each word is considered separately from its surroundings opposed to skip-grams. |
| **Anni** | -Tuning the parameters of the conditional probability function in the neural network (p. 1139).  -Perplexity – recap needed  -step 2 of chapter 2 – are conditional probabilities learnt from the words or from the feature vectors (in the mapping)?  -Formulas p. 1142, p. 1143...  -Verify my view on the difference between a word vector and a word embedding : a word/phrase/document vector can be basically anything (a term-document matrix, one-hot encoded vector) as long as it represents the word/phrase/document in a numerical form. Whereas a word embedding has to come from a vector space, meaning that the embedding already has innate semantic and syntactic information encoded in it and it is more dense than a simple vector.  -Grounded language models and how they  incorporate visual information into generation tasks.  -What actual problem does the grounding solve in neural networks?  -”the model should capture how each constituent in the composed phrase relates to  some perceptual representations.” | -Main problem: so much data that the model will inevitably encounter unseen sentences during testing. Solution: make the model learn distributed representations for words, which inform the model about *semantically similar* sentences. This will allow generalization!  -Statistical models rely on conditional probability.  -Solution to the problem of exponential dimensionality: *associate each word with a feature vector*, *calculate joint probability from the vectors, learn simultaneously both the vectors and the parameters of the probability function.*  -When we manage to train vectors well, we should be able to perform simple algebraic calculations on them to retrieve vectors for a word (Mikolov et al. 2013. Pp. 5) | -**syntagmatic and paradigmatic relations :** words that co-occur vs. words that can be substituted for each other without loss in meaning ?  **-one-hot vector representation :** sparse, doesn’t consider word relations, similarity, basically doesn’t work for semantics? **dense embeddings :** dense vectors that represent the different features / aspects of a word’s meaning  **-word2vec :** for learning word embeddings (semantically relevant) from a corpus. Consists of two main architectures:  **-CBOW :** considers the context of a word (unlike bag-of-words). Aims to predict the target word given the context (words before and after the target word)  **-skip-gram :** Aims to predict the context words given the target word.  -**Gensim :** a library that has many algorithms, for finding semantic patterns and structures from documents.  -**GloVe :** relies on word-word co-occurrence matrices  -**LSTM** : ?  -word embeddings are great for many NLP tasks thanks to their ability to capture both semantic and syntactic meaning. They are also usually computationally more efficient than large vectors. However, we might lose some meaning when creating them (since they are dense(?). Also, not all words have embeddings.  -We can exploit word embeddings when training a probabilistic language model. This way we don’t have to start the feature extraction(?) from scratch.  **-What is the difference between distributional word vectors/matrices, word vectors with dimensionality reduction and word embeddings?**  Word vectors are often more sparse and thus computationally more expensive (?). Word vectors with dimansionality reduction have gone through for instance SVD, with the aim of reducing the dimensions while retaining the most relevant ones. Word embeddings are more dense and have been obtained from neural networks (?).  - “The qualitative human evaluation of how newly composed colour words by this model refer to the colour space suggest that language models can capture  compositionality through gradient learning used with neural networks”.(Ghanimifard & Dobnik, 2017).  -Yes, neural networks can be seen as compositional models (they learn meanings from representations and build up their knowledge of bigger units by looking at how the smaller units combine). |  |

**Answers to Questions / Brainstorming / Questions / discussion from the meeting for seminar 3:**

**syntagmatic and paradigmatic relations, one-hot vector representation, dense embeddings, word2vec, CBOW, skip-gram, Gensim, GloVe, LSTM**

**-** Syntagmatic relations: words that co-occur, paradigmatic relations: words that can be substituted for each other without loss in meaning(?). One-hot-vector representations are quick and easy to make, but they don’t carry any syntactic or semantic meaning, thus not ideal for working with semantics. Dense word embeddings, such as word2vec or GloVe, capture both syntactic and semantic relationships between words in a continuous vector space, which makes them more suitable for tasks involving NLP. When combined with word embeddings, LSTM can utilize the semantic and syntactic information encoded in the embeddings to enhance their performance in tasks.

**-What is the difference between distributional word vectors/matrices, word vectors with dimensionality reduction and word embeddings?**

**-** Word vectors are often more sparse and thus computationally more expensive. Word vectors with dimensionality reduction have gone through for instance SVD, with the aim of reducing the dimensions while retaining the most relevant ones. Word embeddings are denser and have been obtained from neural networks.

**-What are the benefits and weaknesses of using word embeddings compared to other word representations in the previous point - both in terms of the nature of representation and computational cost for building and using them?**

-Distributional vectors:

Benefits : ? Distributional vectors encode syntactic relationships such as word order and context, which provides valuable information for tasks like POS.

Drawbacks : very sparse , and many dimensions -> more expensive, less practical

- Dimensionality reduction vectors:

Benefits: can be efficient

Drawbacks: loss of meaning, hard to handle oov

Benefits of word embeddings : Capturing Semantics and solving the curse of dimensionality / less sparse , generalise better, easy to perform simple algebraic computations on embeddings (if they are well trained!)

Drawbacks: difficulties with words that don’t exist in the training and dont have embeddings (PROBLEM ASSIGNMENT 2), bias data dependent

**-What kind of word embeddings can we build; what are differences between?**

GloVe , Word2Vec( skipgram and CBOW)

**-What is a probabilistic language model? How do word embeddings relate to a probabilistic language model?**

A probabilistic language model aims to estimate the probability of a sequence of words in a language. Word embeddings play a crucial role in probabilistic language models by providing dense, continuous representation of words that capture semantic relationships. These embeddings help the model understand the context and meaning of words in a given sequence, which improves the model’s ability to predict the next word (or generate coherent text).

**-Can compositionality be learned from data?**

Compositionality / additive compositionality (mikolov example ), if “Volga River” appears frequently in the same sentence together with the words “Russian” and “river”, the sum of these two word vectors will result in such a feature vector that is close to the vector of “Volga River”.

**-Why do we ground our language model in perceptual informations/locations?**

Grounding a language model can help enhance contextual understanding and the meaning of words in general. It can be used to achieve compositionality.By conditioning the model on perceptual representations, it gains a better understanding of how each part of a phrase relates to the corresponding perceptual features. This grounding enables the model to produce precise descriptions of spatial scenes and improves its capacity to handle novel composed and decomposed descriptions.

**-To what degree are distributional composed representations interpretable?**

Distributionally composed representations aren’t the easiest to interpret as they are sometimes created with very complex models, dimensionality and density also make it harder. We could perform some basic algebraic functions like subtract Russia from Volga River and add UK and expect to get Thames River.

**-Are the representations that we have learned the same as those as we would expect from compositional rules?**

Statistical learning is very different in the sense that there are no linguistic rules such as syntactic structure but mathematical dependencies. Maybe they are similar in that they both try to capture distributional and semantic rules.

**-Neural networks are also compositional models in the sense that they are composed of units and each of these units defines some representations. Do you agree?**

-Yes, neural networks can be seen as compositional models (they learn meanings from representations and build up their knowledge of bigger units by looking at how the smaller units combine).

**-To what degree can we say that a neural network has learned compositional functions like those in formal semantics?**

Neural networks have the capability to learn compositional functions to a certain extent, but their alignment with formal semantics can vary. These networks are proficient at capturing patterns and relationships in data by training on extensive datasets, which allows them to understand compositional structures in language and other domains. However, the learned functions may not always precisely adhere to the rules and principles of formal semantics.

**General comments:**

The bag-of-words technique doesn’t seem to incorporate a lot of compositionality as each word is considered separately from its surroundings opposed to skip-grams.

**What we were discussing in particular / would like to know more about:**

-Why do we need visual representation for the data or, what is the concrete benefit of working with spatial templates? For instance, if the model learns linguistic patterns like “the book is on the table”, “the cat is under the table” from the training data, won’t the model automatically learn that objects like this are either on or under the table (since there won’t be any linguistic patterns matching “inside the table”)? Are spatial templates used to match visual data like images? How does grounding enhance models’ performance dealing with compositionality?

-In what kind of situation would we (if ever) opt to use sparse embeddings? Do we use them for tf-idf vectors/matrices?

-What kind of embeddings do the state-of-the-art language models like ChatGTP use?

-What is the actual difference between distributional vs. distributed semantics?

-How can we deal with words not having embeddings? For instance in an FFNN aimed to do POS-tagging, how do we create the embedding from scratch?

-Other than embeddings always coming from neural networks and containing more detailed semantic information, how do they differ from distributional word vectors? Is the context window size always smaller than with data that was used to create embeddings? Does linearity have something to do with their differences?

**Main points:**

Mainly talked about grounding neural networks and how they work in practice, differences between embeddings obtained from neural network and distributional word vectors and other embedding types, usability of sparse embeddings/vectors. How and if neural networks are compositional. Problematic to deal with oov words. How interpretable are these representations. Formal semantics vs neural networks