Comparison of dependency and constituency grammar with syntax-enhanced transformers
Vergleich von Dependenz- und Konstituenzgrammatik bei syntaxangereicherten Transformern

Caren Daniel

Masterarbeit
im Rahmen des Studiums
Informatik: Intelligente Systeme

14. November 2022

2022-11-14

Comparison of dependency and constituency grammar with syntax-enhanced transformers

Comparison of dependency and constituency grammar
with syntax-enhanced transformers
Vergleich von Dependenz- und Konstituenzgrammatik
bei syntaxangereicherten Transformern

Caren Daniel

Masterarbeit
im Rahmen des Studiums
Informatik: Intelligente Systeme

14. November 2022

2022-11-14

Comparison of dependency and constituency grammar with syntax-enhanced transformers

1 Introduction
2 Basics
3 Task
4 Research Questions
5 Model Architecture
6 Results
7 Conclusion

Introduction

- NLP systems such as BERT can learn some syntactic information from text data but not all (Luo (2021))
- Idea: Leveraging already available syntactic information by using linguistic syntax graphs
  - Dependency grammar style
  - Constituency grammar style
- Graphs can be injected in various ways
  - By modifying BERT to also use some information from the syntax graphs (Bai et al. (2021), Zhang et al. (2020))
  - New approach: by using Graph Neural Networks (GNNs) in conjunction with BERT (Sachan et al. (2021))

Introduction

- NLP systems such as BERT can learn some syntactic information from text data but not all (Luo (2021))
- Idea: Leveraging already available syntactic information by using linguistic syntax graphs
  - Dependency grammar style
  - Constituency grammar style
- Graphs can be injected in various ways
  - By modifying BERT to also use some information from the syntax graphs (Bai et al. (2021), Zhang et al. (2020))
  - New approach: by using Graph Neural Networks (GNNs) in conjunction with BERT (Sachan et al. (2021))

- BERT: Machine Learning System für eine Vielzahl von Natural Language Processing-Aufgaben
- Zwei verschiedene Typen von Syntaxgraphen, die im Folgenden noch näher vorgestellt werden
- GNN: relativ neue Entwicklung. Ermöglichen das Extrahieren relevanter Information unter Berücksichtigung der Graphstruktur
- das ist auch womit ich mich in der Arbeit beschäftigt habe

BASICS

- Constituency Grammar
- Dependency Grammar
- BERT
- Graph Neural Networks

BASICS

- Constituency Grammar
- Dependency Grammar
- BERT
- Graph Neural Networks

Constituency Grammar

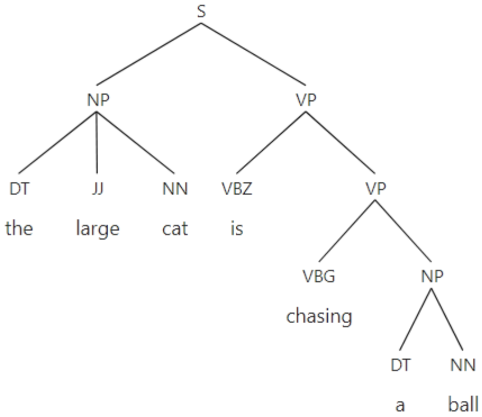
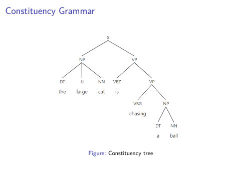


Figure: Constituency tree



- Abbildung zeigt Beispiel eines Constituency Tree
- Satz besteht aus einzelnen Konstituenten (auch: Phrasen)
- Konstituente besteht aus ihrem Kopf und weiteren Elementen, die vom Kopf dominiert werden
- Noun Phrase -> Kopf cat dominiert Artikel (DET) the und Adjektiv (JJ) large
- Konstituente kann auch weitere Konstituenten beinhalten, siehe VP

Dependency Grammar

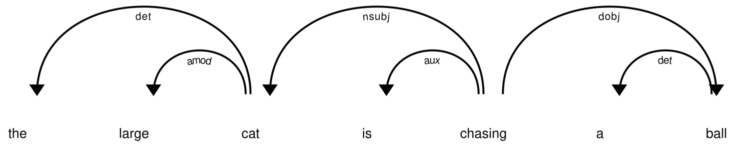
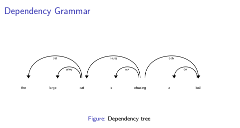


Figure: Dependency tree



- Dependency Grammar hat keine Konstituenten
- stattdessen: Dependency Relations zwischen den Wörtern
- bestehend aus Kopf-Wort, abhängigem Wort und ihrer Relation
- Kopf cat und large: amod-Relation, Kopf cat und the: det-Relation
- kompakter als Constituency Grammar da nur Knoten für die Wörter
- Fokus auf grammatikalischen Relationen

BERT

- BERT: Bidirectional Encoder Representations from Transformers
- Machine-learning model based on transformer architecture
- Transformer is pre-trained on special tasks:
  - **Masked Language Modelling**:
    Reconstruction of masked words in a sequence
  - **Next Sentence Prediction:**
    Prediction whether two sentences follow each other or not
- Can be fine-tuned for a variety of NLP tasks such as NER

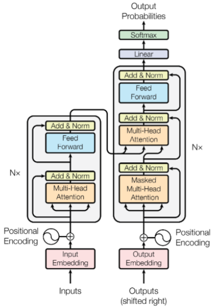
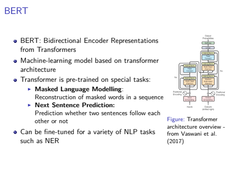


Figure: Transformer architecture overview - from Vaswani et al. (2017)



- Masked Language Modelling: Rekonstruktion maskierter Wörter in einer Sequenz
- Next Sentence Prediction: Vorhersage, ob zwei Sätze aufeinander folgen oder nicht

Graph Neural Networks

- GNNs are able to process and extract information from graph structures
- **Spatial graph convolutional networks**: use spatial information in the graph (e.g. neighbourhood relationships)
- **Message Passing**: aggregate features from all neighbouring nodes into central node
- **Graph Attention Networks** (GAT): selectively aggregate features from neighbours into node by introducing attention weights

Graph Neural Networks
- GNNs are able to process and extract information from graph structures
- **Spatial graph convolutional networks**: use spatial information in the graph (e.g. neighbourhood relationships)
- **Message Passing**: aggregate features from all neighbouring nodes into central node
- **Graph Attention Networks** (GAT): selectively aggregate features from neighbours into node by introducing attention weights

- Graph Neural Networks können Graphstrukturen verarbeiten und auch die Struktur berücksichtigen, um Information zu extrahieren
- Message Passing: Graphstruktur wird genutzt, um auch Informationen über Nachbarknoten in einem Knoten zu sammeln. Dafür werden die Node Features aller Nachbarknoten und des Knotens selbst aggregiert.
- Nachteil: es kann nicht nach Relevanz selektiert werden. Viele Nachbarn können Ergebnis verwässern.
- Graph Attention: Einführung von attention weights, um selektiv Informationen von bestimmten Knoten übernehmen zu können. Dieses Prinzip wird auch in der Arbeit verwendet

Task

- Create a model using BERT coupled with a GNN processing syntax graphs
- Evaluate performance using dependency and constituency-style graphs
- Choose and process datasets for use with BERT and SynGNN

Task

- Create a model using BERT coupled with a GNN processing syntax graphs
- Evaluate performance using dependency and constituency-style graphs
- Choose and process datasets for use with BERT and SynGNN

Evaluation: Named Entity Recognition

- Popular preprocessing task
- Locate and classify named entities in a text
- NE tags generated by Flair (Akbik et al. (2019))
- 18 NE tags and O denoting a non-NE word

John	W.	Smith	is	turning	thirty
B-PERSON	I-PERSON	E-PERSON	O	O	S-CARDINAL

Evaluation: Named Entity Recognition
- Popular preprocessing task
- Locate and classify named entities in a text
- NE tags generated by Flair (Akbik et al. (2019))
- 18 NE tags and O denoting a non-NE word
John W. Smith is turning thirty
B-PERSON I-PERSON E-PERSON O O S-CARDINAL

- z.B. PERSON, ORG, DATE, EVENT, LANGUAGE, PRODUCT...
- Eigennamenerkennung: weit verbreitete Aufgabe für viele NLP-Systeme
- Flair: Vortrainiertes Modell auf der Basis von BERT zur Named Entity Recognition
- implementiert war auch eine weitere Evaluationsaufgabe, die Experimente konnten aber aufgrund von Zeitmangel nicht mehr durchgeführt werden
- Masked Language Modelling (Wiederherstellen maskierter Wörter, eine der Trainingsaufgaben von BERT)
- Vorteil hier: die Daten sind vorhanden und müssen nicht generiert werden

Research Questions

- Comparison of BERT baseline model and BERT with SynGNN
  - baseline: BERT model trained on only text data
- Comparison of gold-standard and generated syntax graphs
  - gold-standard: datasets with hand-annotated syntax graphs
  - generated: syntax graphs generated by spaCy and Berkeley Neural Parser
- Comparison of SynGNN with constituency and dependency graphs

Research Questions

- Comparison of BERT baseline model and BERT with SynGNN
  - baseline: BERT model trained on only text data
- Comparison of gold-standard and generated syntax graphs
  - gold-standard: datasets with hand-annotated syntax graphs
  - generated: syntax graphs generated by spaCy and Berkeley Neural Parser
- Comparison of SynGNN with constituency and dependency graphs

MODEL ARCHITECTURE

- Overview
- SynGNN
- Highway Gate
- Deviations from Sachan et al.

MODEL ARCHITECTURE

- Overview
- SynGNN
- Highway Gate
- Deviations from Sachan et al.

Model Architecture - Overview

- Model architecture adapted from Sachan et al. (2021)
- **BERT Model**: produces BERT embeddings as output
- **Syntax-GNN**: processes syntax graphs with added BERT embeddings
- **Highway Gate**: combines original BERT embeddings with Syntax-GNN output
- **Task-Specific Classifier**: classifies samples, e.g. by NE tag

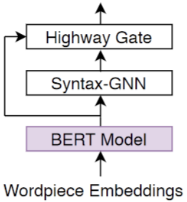


Figure: Model architecture overview - Reproduced from Sachan et al. (2021)

Model Architecture - Overview
- Model architecture adapted from Sachan et al. (2021)
- **BERT Model**: produces BERT embeddings as output
- **Syntax-GNN**: processes syntax graphs with added BERT embeddings
- **Highway Gate**: combines original BERT embeddings with Syntax-GNN output
- **Task-Specific Classifier**: classifies samples, e.g. by NE tag
Figure: Model architecture overview - Reproduced from Sachan et al. (2021)

SynGNN

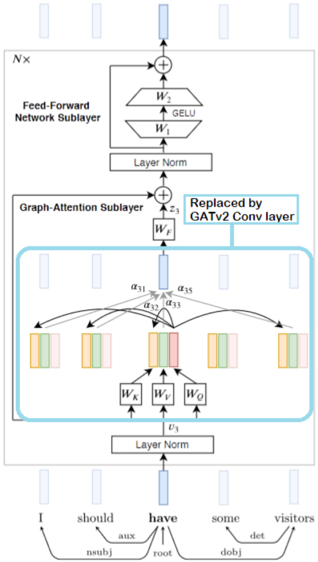
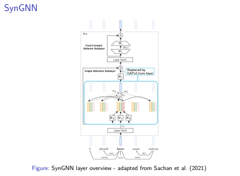


Figure: SynGNN layer overview - adapted from Sachan et al. (2021)



- SynGNN: Architektur eines Transformer Encoders, bei dem die self-attention layer ausgetauscht wurde durch Graph Attention
- Eingabe: Node Features, Edge Features und Liste d. Kanten
- es folgt eine GATv2Conv Layer von Pytorch Geometric
- Residual Connection von: SynGNN Input und Ausgabe des GAT
- Feed-Forward Sublayer: zwei hintereinandergeschaltete lineare Layer sowie Residual Conn. mit deren Output

SynGNN - GATv2Conv Layer

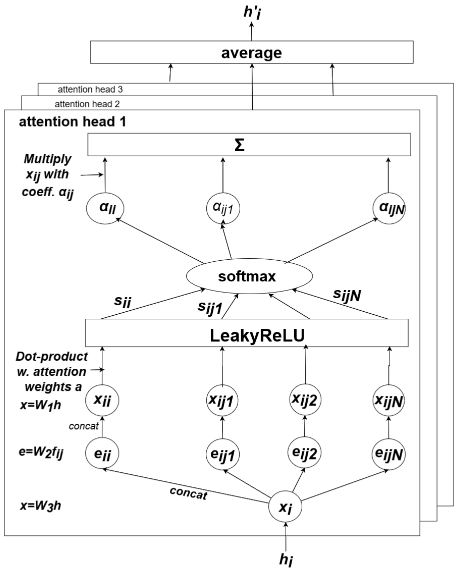
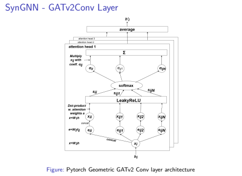


Figure: Pytorch Geometric GATv2 Conv layer architecture



- GATv2 Layer von Pytorch Geometric
- für jede Kante werden die Node Features der beiden Knoten sowie die Edge Features konkateniert
- danach kommen die gemeinsamen attention weights hinzu indem das Skalarprodukt gebildet wird
- die Ergebnisse laufen durch eine LeakyReLU-Aktivierungsfunktion und softmax
- das ergibt die attention scores (= Gewichte mit denen die Node Features berücksichtigt werden)
- Node Features der Nachbarn werden dann mit diesen attention scores multipliziert und aufsummiert
- über Ergebnisse aller attention heads wird der Durchschnitt gebildet

Highway Gate

- Highway Gate as introduced in the paper *Highway Networks* by Srivastava et al. (2015)
- Combines the original BERT embeddings v_i with the SynGNN output z_i
- Allows the model to focus more on the BERT embeddings or the SynGNN output when useful for the task

$$g_i = \sigma(\mathbf{W_g} v_i + b_g)$$
$$h_i = g_i \odot v_i + (1 - g_i) \odot z_i$$

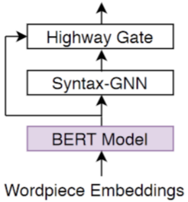
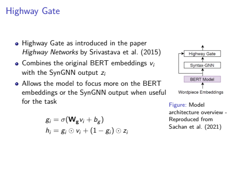


Figure: Model architecture overview - Reproduced from Sachan et al. (2021)



- Highway Gate aus Paper Highway Networks
- Erlaubt es dem Modell, BERT embeddings oder SynGNN Output weiterzugeben
- Lineare Layer mit sigmoid-Aktivierungsfunktion und BERT embeddings als Input (=Wert zwischen 0/1 für jedes token)
- Ergebnis bestimmt ob BERT embeddings, SynGNN-Output oder Mischung aus beiden weitergegeben werden

Deviations from Sachan et al.

- Replaced original GAT algorithm with GATv2Conv layer
  - Inclusion of edge features (=dependency relations) in the calculations
  - Separate learnable weights for node features, neighbour node features and edge features
  - Introduction of self-loops for each node
- Use of Cross-Entropy loss function (no loss function specified in paper)
- Use of class weights to counteract imbalanced class distribution

Deviations from Sachan et al.

- Replaced original GAT algorithm with GATv2Conv layer
  - Inclusion of edge features (=dependency relations) in the calculations
  - Separate learnable weights for node features, neighbour node features and edge features
  - Introduction of self-loops for each node
- Use of Cross-Entropy loss function (no loss function specified in paper)
- Use of class weights to counteract imbalanced class distribution

RESULTS

- Overview
- Comparison of BERT baseline and SynGNN
- Comparison of Gold-standard and Generated Syntax Graphs
- Constituency vs Dependency SynGNN

RESULTS

- Overview
- Comparison of BERT baseline and SynGNN
- Comparison of Gold-standard and Generated Syntax Graphs
- Constituency vs Dependency SynGNN

Results - Overview

Model	Dependency Prec.	Dependency Recall	Dependency F1	Constituency Prec.	Constituency Recall	Constituency F1
BERT Cased	0.68	0.73	0.70	0.62	0.65	0.63
SynGNN Gold Cased	0.70	0.87	0.77	0.66	0.90	0.76
SynGNN Gen. Cased	-	-	-	0.66	0.89	0.76
BERT Uncased	0.67	0.71	0.69	0.60	0.59	0.60
SynGNN Gold Uncased	0.63	0.86	0.72	0.56	0.88	0.69
SynGNN Gen. Uncased	-	-	-	0.57	0.89	0.69

Table: Overview of NER results for BERT baseline and SynGNN on full datasets

Results - Overview

Model	Dependency Prec.	Dependency Recall	Dependency F1	Constituency Prec.	Constituency Recall	Constituency F1
BERT Cased	0.68	0.73	0.70	0.62	0.65	0.63
SynGNN Gold Cased	0.70	0.87	0.77	0.66	0.90	0.76
SynGNN Gen. Cased	-	-	-	0.66	0.89	0.76
BERT Uncased	0.67	0.71	0.69	0.60	0.59	0.60
SynGNN Gold Uncased	0.63	0.86	0.72	0.56	0.88	0.69
SynGNN Gen. Uncased	-	-	-	0.57	0.89	0.69

Table: Overview of NER results for BERT baseline and SynGNN on full datasets

- Ergebnisse der Experimente auf den kompletten Datensätzen
- Unterteilt in zwei Kategorien: Cased/Uncased d.h. welcher BERT tokenizer benutzt wurde (mit GroSSbuchstaben/Umwandlung in Kleinbuchstaben)
- Precision: Anteil richtig klassifizierter Elemente einer Klasse ggü. allen die als die Klasse eingeordnet wurden inkl. False Positives aus anderen Klassen
- Recall: Anteil Elemente, die richtig klassifiziert wurden ggü. allen Elementen der Klasse
- F1: harmonisches Mittel von Precision und Recall
- Diskrepanz Prec./Recall erklärt sich durch falsche Zuordnung von Nicht-NEs (O) zu NE Tags

Comparison of BERT baseline and SynGNN

- BERT Cased: BERT+SynGNN performs better than baseline model for both types of graphs
- Constituency SynGNN improves more over its baseline model than dependency SynGNN
- BERT Uncased: BERT+SynGNN again perform better than the baseline, but to a lesser degree

	Dependency			Constituency		
Model	Prec.	Recall	F1	Prec.	Recall	F1
BERT Cased	0.68	0.73	0.70	0.62	0.65	0.63
SynGNN Gold Cased	0.70	0.87	0.77	0.66	0.90	0.76
BERT Uncased	0.67	0.71	0.69	0.60	0.59	0.60
SynGNN Gold Uncased	0.63	0.86	0.72	0.56	0.88	0.69

Table: Overview of NER results for BERT baseline and SynGNN on full datasets

Comparison of BERT baseline and SynGNN
- BERT Cased: BERT+SynGNN performs better than baseline model for both types of graphs
- Constituency SynGNN improves more over its baseline model than dependency SynGNN
- BERT Uncased: BERT+SynGNN again perform better than the baseline, but to a lesser degree
Table: Overview of NER results for BERT baseline and SynGNN on full datasets

- Cased: BERT + SynGNN schneidet für beide Typen von Syntaxgraphen besser ab als das Baseline Modell mit nur BERT
- Dependency SynGNN: F1 0.7 auf 0.77 - Constituency SynGNN: 0.63 auf 0.76
- Verbesserung beim SynGNN + Constituency Graphs um 13 Punkte, bei Dependency nur 7
- Dabei bleibt Precision ähnlich, Recall erhöht sich von 0.65 auf 0.9, also 90% aller NEs werden richtig zugeordnet
- Uncased: Verbesserung Baseline zu SynGNN nicht mehr so sichtbar (Baseline-Performance bricht ein)
- Dep: F1 0.69 auf 0.72 - Const: F1 0.6 auf 0.69

Comparison of Gold-standard and Generated Syntax Graphs

- Syntax graphs were generated for constituency graphs only
- Performance of SynGNN with gold and generated trees is almost identical
- Cased: same F1 of 0.76
- Uncased: same F1 of 0.69

	Constituency		
Model	Prec.	Recall	F1
SynGNN Gold Cased	0.66	0.90	0.76
SynGNN Gen. Cased	0.66	0.89	0.76
SynGNN Gold Uncased	0.56	0.88	0.69
SynGNN Gen. Uncased	0.57	0.89	0.69

Table: Overview of NER results for SynGNN with gold-standard and generated constituency graphs

Comparison of Gold-standard and Generated Syntax Graphs
- Syntax graphs were generated for constituency graphs only
- Performance of SynGNN with gold and generated trees is almost identical
- Cased: same F1 of 0.76
- Uncased: same F1 of 0.69

	Constituency		
Model	Prec.	Recall	F1
SynGNN Gold Cased	0.66	0.90	0.76
SynGNN Gen. Cased	0.66	0.89	0.76
SynGNN Gold Uncased	0.56	0.88	0.69
SynGNN Gen. Uncased	0.57	0.89	0.69

Table: Overview of NER results for SynGNN with gold-standard and generated constituency graphs

- Ergebnisse nur für SynGNN mit Constituency Graphen: Syntaxgraphen wurden nur für Constituency Grammar erzeugt, nicht für Dependency
- Ergebnisse für die handerstellten und generierten Graphen sind beim F1 identisch
- Hinweis, dass sich die Graphen durch gute generierte ersetzen lassen

Constituency vs Dependency SynGNN

- Original datasets were imbalanced in size. The comparison was made on 'balanced' datasets with approx. the same number of sentences and NE tags in both sets

	Dependency			Constituency		
Model	Prec.	Recall	F1	Prec.	Recall	F1
SynGNN Gold Cased Bal.	0.60	0.85	0.70	0.66	0.90	0.76
SynGNN Gold Uncased Bal.	0.57	0.84	0.68	0.56	0.88	0.69

Table: Overview of NER results for BERT baseline and SynGNN on balanced datasets

Constituency vs Dependency SynGNN
Original datasets were imbalanced in size. The comparison was made on 'balanced' datasets with approx. the same number of sentences and NE tags in both sets
Table: Overview of NER results for BERT baseline and SynGNN on balanced datasets

- Vergleich von SynGNN mit Constituency Graphen und Dependency Graphen
- Dafür mussten die Originaldatensätze angepasst werden
- stimmen nicht in der Anzahl Sätzen bzw. NE tags überein
- 'balanced' Datensätze: wurden so ausgewählt, dass diese ungefähr übereinstimmen
- Cased: SynGNN mit Konstituenzgraphen schneidet besser ab. Dep F1 0.7 - Const F1 0.76
- Uncased: Unterschied ist hier nur noch marginal. Dep F1 0.68 - Const F1 0.69

Comparison with Selected NE Tags

- All NE tags where F1 value $>$ 0.5 and the difference in F1 of dep. and const. results is more than 10 points
- Using constituency graphs, the scores are significantly higher

	Dependency					Constituency			
NE type	Prec.	Recall	F1	#	NE type	Prec.	Recall	F1	#
CARDINAL	0.60	0.85	0.70	301	CARDINAL	0.74	0.93	0.82	493
FAC	0.26	0.65	0.37	69	FAC	0.45	0.68	0.54	66
LANGUAGE	0.53	0.60	0.56	47	LANGUAGE	0.81	0.93	0.87	14
ORG	0.51	0.80	0.62	412	ORG	0.63	0.88	0.74	468
QUANTITY	0.28	0.61	0.39	31	QUANTITY	0.41	0.81	0.55	101

Table: Selected NER results for SynGNN gold with cased BERT for balanced dependency and constituency dataset.

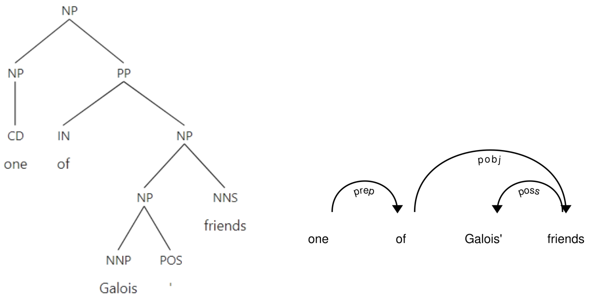
Comparison with Selected NE Tags

- All NE tags where F1 value $>$ 0.5 and the difference in F1 of dep. and const. results is more than 10 points
- Using constituency graphs, the scores are significantly higher

Table: Selected NER results for SynGNN gold with cased BERT for balanced dependency and constituency dataset.

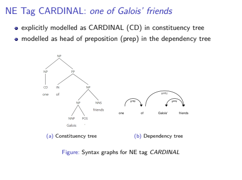
NE Tag CARDINAL: *one of Galois' friends*

- explicitly modelled as CARDINAL (CD) in constituency tree
- modelled as head of preposition (prep) in the dependency tree



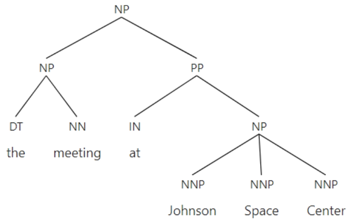
(a) Constituency tree (b) Dependency tree

Figure: Syntax graphs for NE tag *CARDINAL*

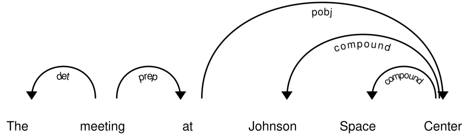


- F1 Dependency Grammar: 0.7
- F1 Constituency Grammar: 0.82
- Verbesserung in Precision und Recall
- Recall bei Constituency Gr. sehr hoch mit 0.93 aller CARDINALs richtig klassifiziert
- nicht überraschend, da *one* im constituency tree als CARDINAL markiert ist, was dem Modell bei der Bestimmung helfen dürfte
- im Dependency tree ist für *one* nur die grammatikalische Funktion als Präposition modelliert, die Eigenschaft als Kardinalzahl fällt weg

NE Tag FAC: *the meeting at Johnson Space Center*

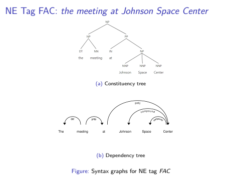


(a) Constituency tree



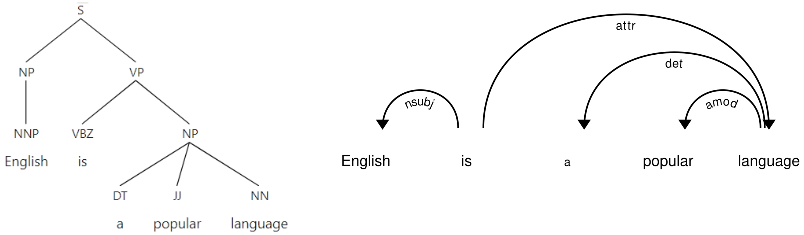
(b) Dependency tree

Figure: Syntax graphs for NE tag *FAC*



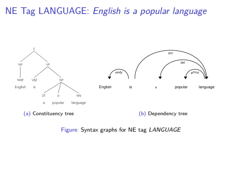
- F1 Dependency: 0.37 F1 Constituency: 0.54
- Recall ist fast gleich (0.65 -> 0.68) nur Precision verbessert sich von 0.26 zu 0.45
- d.h. die beiden Syntaxtypen werden ungefähr dieselbe Anzahl an FAC-Entitäten korrekt klassifiziert
- aber: das SynGNN mit den Dependency Graphen klassifiziert mehr andere NEs fälschlich als Gebäudenamen
- die FAC ist im Constituency tree als NP-Knoten modelliert, der drei NNP-Knoten dominiert (=Proper Noun/Eigenname)
- im Dependency Tree: *Center* ist Kopf zweier compound-Knoten. Compound wird nur für zusammengesetzte Nomen verwendet, weist also auf die Wortart hin. Daher Recall ähnlich
- Unterschied in der Precision: die Informationen im Dep. Tree sind nicht spezifisch genug und es muss noch eine andere Konstruktion mit ähnlichem Syntaxbaum-Aufbau geben, die ebenfalls als FAC klassifiziert wird

NE Tag LANGUAGE: *English is a popular language*



(a) Constituency tree (b) Dependency tree

Figure: Syntax graphs for NE tag *LANGUAGE*



- grösste Abweichung mit F1 0.56 zu 0.87 beim SynGNN mit Constituency Graphen
- Im Constituency Tree: modelliert als NNP (=Eigenname)
- im Dependency Tree: wieder Fokus auf gramm. Funktion mit nsubj (= Subjekt von is)
- sehr häufig und nicht spezifisch genug, was die schlechtere Performance erklären könnte
- auch: mehr LANGUAGE Entitäten zu klassifizieren: 47 bei Dep., 14 bei Const.

Conclusion

- In all experiments, the SynGNN performs consistently better than BERT only
- SynGNN with constituency graphs seem to perform better than with dependency graphs
- For constituency graphs, performance of gold and generated graphs is similar
- Sachan et al. (2021): No performance gain over BERT with dependency graphs and NER task

Conclusion
- In all experiments, the SynGNN performs consistently better than BERT only
- SynGNN with constituency graphs seem to perform better than with dependency graphs
- For constituency graphs, performance of gold and generated graphs is similar
- Sachan et al. (2021): No performance gain over BERT with dependency graphs and NER task

- In allen Experimenten schneidet das SynGNN Modell besser ab als das Baseline Modell nur mit BERT
- Constituency Graphen scheinen für NER besser geeignet zu sein als Dependency
- Für Constituency Graphen schneiden generierte und gold-standard Graphen ähnlich ab
- Ergebnis weicht ab von dem, das Sachan et al. mit NER erreichen. Sie stellten keine signifikante Verbesserung ggü. BERT fest mit Dependency Graphen
- Gründe: Fehlen von relevanten Infos, da sie die Kanten-Features (=Relationen) nicht berücksichtigen
- GröSSerer Datensatz bei Sachan et. al.: ggf. kann BERT so alle relevanten Informationen selbst lernen. Kleine Datensätze profitieren ggf. von Syntaxgraphen

Future Work

- Evaluate performance with generated dependency graphs
- Different evaluation tasks (Masked Language Modeling, Semantic Role Labeling, Relation Extraction)
- Development of model with automatic generation of syntax graphs for any text

Future Work

- Evaluate performance with generated dependency graphs
- Different evaluation tasks (Masked Language Modeling, Semantic Role Labeling, Relation Extraction)
- Development of model with automatic generation of syntax graphs for any text

Thank You

References I

A. Akbik, T. Bergmann, D. Blythe, K. Rasul, S. Schweter, and R. Vollgraf. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59, 2019.

J. Bai, Y. Wang, Y. Chen, Y. Yang, J. Bai, J. Yu, and Y. Tong. Syntax-bert: Improving pre-trained transformers with syntax trees. *CoRR*, abs/2103.04350, 2021. URL `https://arxiv.org/abs/2103.04350`.

Z. Luo. Have attention heads in BERT learned constituency grammar? *CoRR*, abs/2102.07926, 2021. URL `https://arxiv.org/abs/2102.07926`.

References I

A. Akbik, T. Bergmann, D. Blythe, K. Rasul, S. Schweter, and R. Vollgraf. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59, 2019.

J. Bai, Y. Wang, Y. Chen, Y. Yang, J. Bai, J. Yu, and Y. Tong. Syntax-bert: Improving pre-trained transformers with syntax trees. *CoRR*, abs/2103.04350, 2021. URL `https://arxiv.org/abs/2103.04350`.

Z. Luo. Have attention heads in BERT learned constituency grammar? *CoRR*, abs/2102.07926, 2021. URL `https://arxiv.org/abs/2102.07926`.

References II

D. Sachan, Y. Zhang, P. Qi, and W. L. Hamilton. Do syntax trees help pre-trained transformers extract information? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2647–2661, Online, April 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.228. URL https://aclanthology.org/2021.eacl-main.228.

R. K. Srivastava, K. Greff, and J. Schmidhuber. Highway networks. *CoRR*, abs/1505.00387, 2015. URL http://arxiv.org/abs/1505.00387.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL http://arxiv.org/abs/1706.03762.

Z. Zhang, Y., J. Zhou, S. Duan, H. Zhao, and R. Wang. Sg-net: Syntax guided transformer for language representation. *CoRR*, abs/2012.13915, 2020. URL https://arxiv.org/abs/2012.13915.

References II

D. Sachan, Y. Zhang, P. Qi, and W. L. Hamilton. Do syntax trees help pre-trained transformers extract information? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2647–2661, Online, April 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.228. URL https://aclanthology.org/2021.eacl-main.228.

R. K. Srivastava, K. Greff, and J. Schmidhuber. Highway networks. *CoRR*, abs/1505.00387, 2015. URL http://arxiv.org/abs/1505.00387.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL http://arxiv.org/abs/1706.03762.

Z. Zhang, Y., J. Zhou, S. Duan, H. Zhao, and R. Wang. Sg-net: Syntax guided transformer for language representation. *CoRR*, abs/2012.13915, 2020. URL https://arxiv.org/abs/2012.13915.