When and how to paraphrase for named entity recognition?

Saket Sharma[1¶], **Aviral Joshi**[1¶], **Yiyun Zhao**[1¶], **Namrata Mukhija**[1¶],
Hanoz Bhathena[¶], **Prateek Singh**[¶], **Sashank Santhanam**[‖2]

[¶]Machine Learning Center of Excellence, JPMorgan Chase & Co.
[‖]Apple
saket.sharma@jpmchase.com

Abstract

While paraphrasing is a promising approach for data augmentation in classification tasks, its effect on named entity recognition (NER) is not investigated systematically due to the difficulty of span-level label preservation. In this paper, we utilize simple strategies to annotate entity spans in generations and compare established and novel methods of paraphrasing in NLP such as back translation, specialized encoder-decoder models such as Pegasus, and GPT-3 variants for their effectiveness in improving downstream performance for NER across different levels of gold annotations and paraphrasing strength on 5 datasets. We thoroughly explore the influence of paraphrasers, dynamics between paraphrasing strength and gold dataset size on the NER performance with visualizations and statistical testing. We find that the choice of the paraphraser greatly impacts NER performance, with one of the larger GPT-3 variants exceedingly capable of generating high quality paraphrases, yielding statistically significant improvements in NER performance with increasing paraphrasing strength, while other paraphrasers show more mixed results. Additionally, inline auto annotations generated by larger GPT-3 are strictly better than heuristic based annotations. We also find diminishing benefits of paraphrasing as gold annotations increase for most datasets. Furthermore, while most paraphrasers promote entity memorization in NER, the proposed GPT-3 configuration performs most favorably among the compared paraphrasers when tested on unseen entities, with memorization reducing further with paraphrasing strength. Finally, we explore mention replacement using GPT-3, which provides additional benefits over base paraphrasing for specific datasets.

[1]Equal contribution.
[2]Currently at Apple. Work done while at JPMorgan Chase & Co.

1 Introduction

Named entity recognition (NER) seeks to extract entity mentions (e.g., Shakespeare, Warwickshire) from a text (Shakespeare was born and raised in Warwickshire) for predefined categories of interest (such as people and locations). It is a critical component underpinning many industrial pipelines for a variety of downstream natural language processing applications such as search, recommendation, and virtual assistant systems. However, in real-world applications, there is often a scarcity of labeled data for training advanced deep neural models because span-level NER annotations are costly, and domain expertise may be needed to annotate data from domains such as finance, biomedical sciences, etc.

Data augmentation is often used as an alternative to address the data scarcity issue in many NLP tasks (see an NLP data augmentation survey by Feng et al. (2021)). However, data augmentation for NER imposes additional challenges because NER requires token/span level label preservation. Therefore, most existing works on NER data augmentation primarily focus on local replacement for entity mentions (Dai and Adel, 2020; Zhou et al., 2022; Liu et al., 2022; Wenjing et al., 2021) as well as context words (Dai and Adel, 2020; Li et al., 2020). The replacements can be other mentions with the same labels (Dai and Adel, 2020), synonyms from an external lexical resource such as wordnet (Dai and Adel, 2020), or tokens generated by the pretrained language models such as BERT via masked token task (Zhou et al., 2022; Liu et al., 2022; Wenjing et al., 2021). However, to enhance the reliability of masked token prediction, the language model usually needs to be fine-tuned on the NER training data and label information is often inserted close to the [MASK]s (Zhou et al., 2022; Wenjing et al., 2021), which requires a decent amount of labeled training data. A recent study

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 7052–7087

by Ding et al. (2020) trained a sequence generator to synthesize sentences with inline NER annotations that can create novel NER training examples beyond local modifications but requires sufficient NER labeled examples for training the generator.

This work primarily focuses on the less-studied data augmentation method for NER – paraphrasing – which has the potential to introduce structural and lexical replacement and does not assume many labeled examples. Specifically, we compare established, and novel paraphrasing methods and propose simple ways to preserve span-level labels. Unlike most existing studies that focus on the influence of the amount of gold data only, we systematically investigate the effect of different levels of paraphrasing on downstream performance, at different levels of gold annotations across 5 datasets. We investigate the quality of paraphrases from 6 different systems as augmentation data, as well as stand alone training data for NER. We further examine the entity memorization via the performance change on unseen mentions for each entity and address the issue with mention replacement.

We find paraphrasing to be generally effective in low data regimes for most paraphrasers. However, the choice of paraphraser affects the magnitude, and direction of the change in performance across all levels of gold data. We find the use of LLMs to generate inline annotations[1] while paraphrasing to be superior to simpler heuristics, and GPT-3 Davinci variant with inline annotations to be a generally superior choice across datasets for paraphrasing. In addition, our entity level analysis shows that entity classes with low support (number of mentions) or low number proportion benefit more from paraphrasing. We then investigate whether there is an indications of entity memorization with increasing paraphrasing strength, and find that GPT-3 Davinci variant with inline annotations is more robust against entity memorization compared to other paraphrasers. We further reduce memorization in some datasets by introducing mention replacement based on GPT-3 DaVinci in the paraphrasing pipeline.

2 Datasets and Paraphrasers

2.1 Datasets

NER datasets are chosen to have coverage across a variety of domains including news, Wikipedia, Twitter, biomedical research and search; while also having a diverse set of entity types (word phrases, alphanumeric, datetime, alphabetical etc.).

We choose 5 datasets based on the above principles: Ontonotes5 (Hovy et al., 2006), Tweebank (Jiang et al., 2022), WNUT 2017 (Derczynski et al., 2017), MIT Restaurant NER dataset (MIT-R) (Liu et al., 2013), BioCreative V CDR (BC5CDR) (Wei et al., 2016). Pre-formatted versions of all datasets are sourced from the TNER project (Ushio and Camacho-Collados, 2021) on Huggingface datasets (Lhoest et al., 2021) (See Appendix A.16). Datasets such as WNUT also have rare entities by design, allowing us to probe robustness against entity memorization.

	MIT-R	Onto -notes	BC5 -CDR	Twee -bank	Wnut -17
BT	1	0	2	0	5
Pegasus	1	0	13	3	8
Ada-A	10	0	0	11	0
Ada-B	4	0	0	**16**	2
DaV-A	3	0	4	5	3
DaV-B	**26**	**35**	**26**	10	**27**

Table 1: Counts the configurations of G & P where a paraphraser shows highest relative improvement over no paraphrasing baseline for a given G (Division by zero is avoided by using absolute improvement). GPT-3 DaV-B outperforms other paraphrasers across most datasets. Detailed results of downstream performance for all datasets, paraphrasers, configurations can be found in Appendix A.6.

2.2 Paraphrasers and postprocessing

In our experiments, we compare six paraphrasing systems: (1) Back Translation, (2) Pegasus, (3) Ada (Prompt A) / Ada-A, (4) Ada (Prompt B) / Ada-B, (5) Davinci (Prompt A) / DaV-A and (6) Davinci (Prompt B) / DaV-B. We generate a maximum of 4 unique paraphrases per gold sentence for each paraphraser and postprocess the paraphrases with simple re-annotation and filtering.

2.2.1 Back-translation; BT

Back translation has been widely used as a data augmentation method (Sugiyama and Yoshinaga, 2019; Corbeil and Ghadivel, 2020; Xie et al., 2020) including in phrase based systems like (Bojar and Tamchyna, 2011). For our experiments we use pre-trained English-German and German-English models (∼738M parameters) available from Huggingface model hub [2] via Tiedemann and Thot-

[1] Inline annotation: [Shakespeare](PERSON) was born and raised in [Warwickshire](LOC)

[2] https://huggingface.co/models

Dataset	Recall (%)
WNUT-17	93.2
Tweebank	92.9
Ontonotes	83.8
MIT-R	71.4
BC5CDR	90.3

Table 2: Entity recall across datasets for DaV-B without any post processing. Recall is calculated via a case insensitive search, so acts as a lower bound.

	MIT-R	Onto -notes	BC5 -CDR	Twee -bank	Wnut -17
BT	0.66	0.74	0.76	**0.41**	0.30
Pegasus	0.68	0.75	0.78	0.33	0.23
Ada-A	0.71	0.73	0.74	0.36	0.23
Ada-B	0.70	0.72	0.74	0.34	0.23
DaV-A	0.67	0.75	0.76	0.39	0.27
DaV-B	**0.73**	**0.80**	**0.82**	**0.41**	**0.32**

Table 3: Test micro-F1 when training using only paraphrases with P=1 for full dataset. Number in bold is the maximum for a given dataset. GPT-3 DaV-B outperforms all paraphrasers across datasets.

tingal (2020) and the model architecture used is BART (Lewis et al., 2019). We use a temperature parameter of 0.8 with greedy decoding.

2.2.2 PEGASUS Paraphraser

PEGASUS (∼568M parameters), introduced in (Zhang et al., 2020) for the purpose of summarization, is a large pre-trained transformer (Vaswani et al., 2017) based encoder-decoder model, pre-trained using a custom self-supervised objective. To use it as a paraphraser the model was fine-tuned on a paraphrasing task. We use an off-the-shelf version of PEGASUS fine-tuned for paraphrasing released on Huggingface model hub. [3]

2.2.3 GPT-3 variants

GPT-3 (Brown et al., 2020) is an auto-regressive decoder only transformer pre-trained for language modeling, showing impressive in-context learning, and instruction following ability (Radford et al., 2019; Sanh et al., 2021; Wei et al., 2021; Ouyang et al., 2022; Campos and Shern, 2022). We use the OpenAI API [4] to query the text-ada-001 (∼350M parameters), and text-davinci-002 (∼175B parameters) variants of GPT-3. We prompt both GPT-3 variants with two versions of one shot prompts with a temperature of 0.8, max length of 100, and default values for other parameters:

Prompt A GPT-3 variant is instructed to generate paraphrases without specific instruction to retain inline annotation for entities:

" Create a paraphrase for inputs like the following example:

Input: Japanese band The Altruists is releasing their hit single this fall.
Paraphrases:
1. The Altruists, a Japanese band is releasing their hit single this fall

Input: `BLANK`
Paraphrases:
1."

Prompt B GPT-3 variant is instructed to generate paraphrases, while also retaining inline annotation for entities (highlighted in red):

" Create a paraphrase for inputs like the following example. Preserve the annotations in the [] and ():

Input: Japanese band [The Altruists](ORG) is releasing their hit single this fall.
Paraphrases:
1. [The Altruists](ORG), a Japanese band is releasing their hit single this fall

Input: `BLANK`
Paraphrases:
1."

During paraphrasing, "`BLANK`" is replaced by an actual gold sentence being paraphrased.

We conduct light prompt tuning based on entity recall to select Prompt B, (Prompt A is then created by dropping the annotation retention instructions). The prompt that retains annotations for most gold entity mentions (based on case insensitive string match) in generated paraphrases, is chosen as the final prompt. Table 2 shows the raw entity recall for GPT-3 DaV with Prompt B across datasets.

2.2.4 Post-processing & filtering of paraphrases

We re-annotate outputs of all paraphrasers based on a case insensitive exact match search for the entity values present in gold sentence. In the case of LLMs generating inline annotations, this logic is used to supplement annotations generated by the model, relying on the model generated annotations in cases of conflicts. Further filtering is applied to the paraphrases from all models to remove paraphrases for gold sentences shorter than 15 characters, remove paraphrases that are a duplicate of

[3] https://huggingface.co/tuner007/pegasus_paraphrase
[4] https://beta.openai.com/

the gold sentence or of another paraphrase, and when generation contains an entity not present in entity space of the dataset. We also retain only the first generation of multiline generations for paraphrasers generating a numbered list of paraphrases (common with prompt driven GPT-3 variants Appendix A.2).

For each paraphrasing configuration (model + post-processing), we evaluated the entity recall rate of the synthetic data as well as the language quality of 100 examples sampled from each dataset. We find that DaV-B consistently outperforms other paraphrasers in both entity recall and paraphrasing quality metrics (See Appendix A.4).

3 Experiments

3.1 Using gold & paraphrasing data for training NER

3.1.1 Experimental setup

In practical settings, gold training data is generated incrementally. Paraphrases then are created using none/some/all of the gold data which simulates a change in paraphrasing strength. We present results based on different configurations of gold ratio (G-ratio), i.e. what percentage of gold data is used in a particular configuration, and paraphrase ratio (P-ratio), i.e. what is the ratio of number of paraphrases compared to number of gold samples.

Gold Sampling When generating gold sample for G=0.01, we sample 1% of the total dataset, stratified by entities (and an equivalent percentage of gold samples with no entities). Subsequently, moving to G=0.03, we retain the sample from the first step, and sample an additional 2% from the remaining dataset[5]. Experiments are conducted for these G-ratios: 0.01, 0.03, 0.05, 0.07, 0.09, 0.11, 0.25, 0.5, 1.0[6].

Paraphrase Sampling For all G/P ratio configurations, after sampling gold samples using the process above, a random set of paraphrases are then sampled for the gold samples in the set based on the P-ratio. For example, for P=0.25, the number of paraphrasing samples is a fourth of the gold samples used in the configuration.

The following P-ratios are explored for every G-ratio: 0.0 (no paraphrasing), 0.25, 0.5, 1.0, 2.0, 4.0.

For each G/P ratio, the corresponding dataset is used to fine-tune a distilbert-base-cased base (66M parameters) model (Sanh et al., 2019) [7] for named entity recognition using the 1-step training described by (Okimura et al., 2022) using standard classification loss over hidden states of individual tokens. The models are trained with early stopping (patience=5, metric=eval_F1).

We generate overall, and entity specific micro F1 for each G/P combination along with standard deviation across three runs.

3.1.2 Analysis method

We first present visualizations and tables to summarize the general trends of the overall NER F1 performance improvement with respect to different paraphrasers and the dynamics of paraphrase ratio and gold ratio.

To support the observations made from the figures (1, 2) and tables (1, 3) we perform analysis at the entity level, by conducting statistical tests on the downstream performance improvement Δ_{F1}, where

$$\Delta_{F1}(g, p, ent) = F1(g, p, ent) - F1(g, 0, ent)$$

Specifically, we investigate whether the change in downstream NER F1 depends on certain characteristics of an entity including entity support (how many examples one entity class contains) and surface form features (proportion of capitalizations and numbers in entity types). We build a linear regression model using the entity characteristics aforementioned along with the paraphraser, G & P ratios as the predictors and Δ_{F1} as the dependent variable, formalized as follows:

$\Delta_{F1} \sim \mathbf{P}araphrase * (\mathbf{G}old + model + support + \mathbf{cap}italize + number)$

3.1.3 Results

Effect of Paraphrasers Table 1 shows the counts across G&P configurations where a paraphraser has the highest relative improvement and Figure 1 demonstrates the F1 change after adding the synthetic data. Both suggest, the choice of paraphraser strongly dictates the augmentation performance. GPT-3 DaV-B consistently outperforms, or matches other paraphrasers and is a safe default choice for paraphrasing across domains. Across the Davinci variants, inline annotations with Prompt B

[5]This incremental nature of sampling gold data simulates real projects

[6]We only go up to G=0.25 for large Ontonotes dataset for speed

[7]https://huggingface.co/distilbert-base-cased

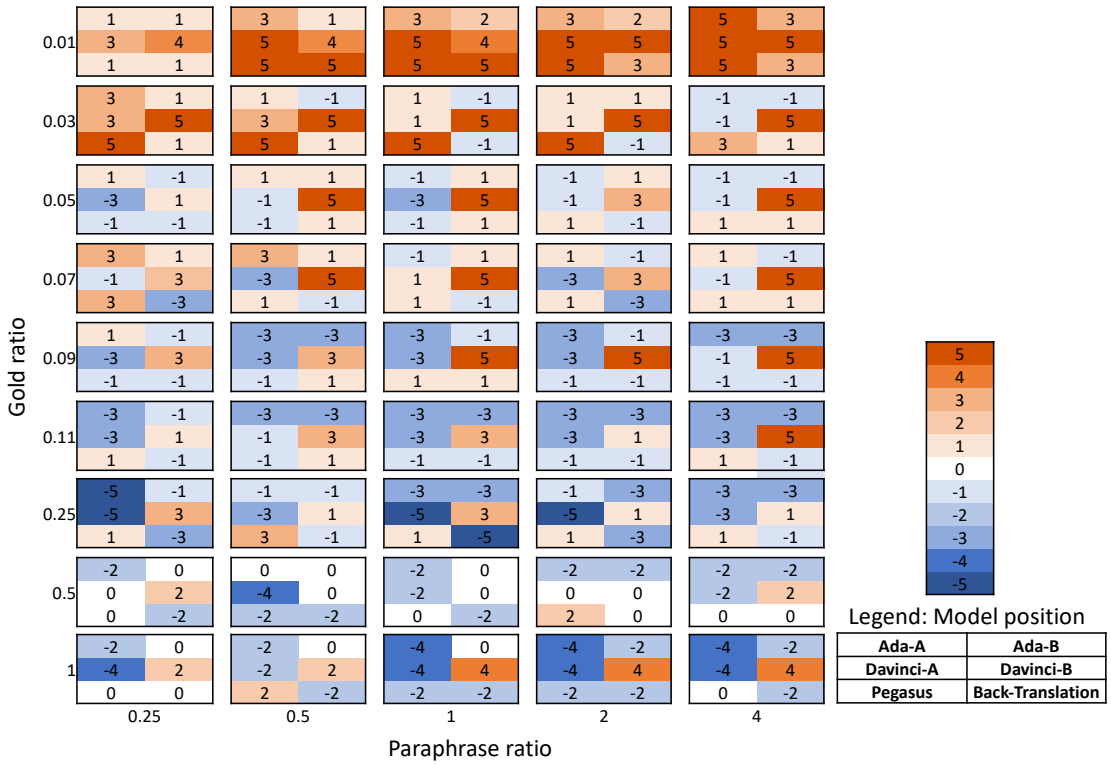


Figure 1: Matrix of scores of how F1 changed relative to the no paraphrasing (P=0) baseline after the addition of synthetic data across datasets for different G & P ratios. Improvement/worsening (shown in color) in any dataset at a given G/P ratio gets a score of +1/-1 respectively, and aggregation is then done across datasets. The model position legend shows the position for each paraphraser (e.g., the upper left cell always corresponds to Ada-A).

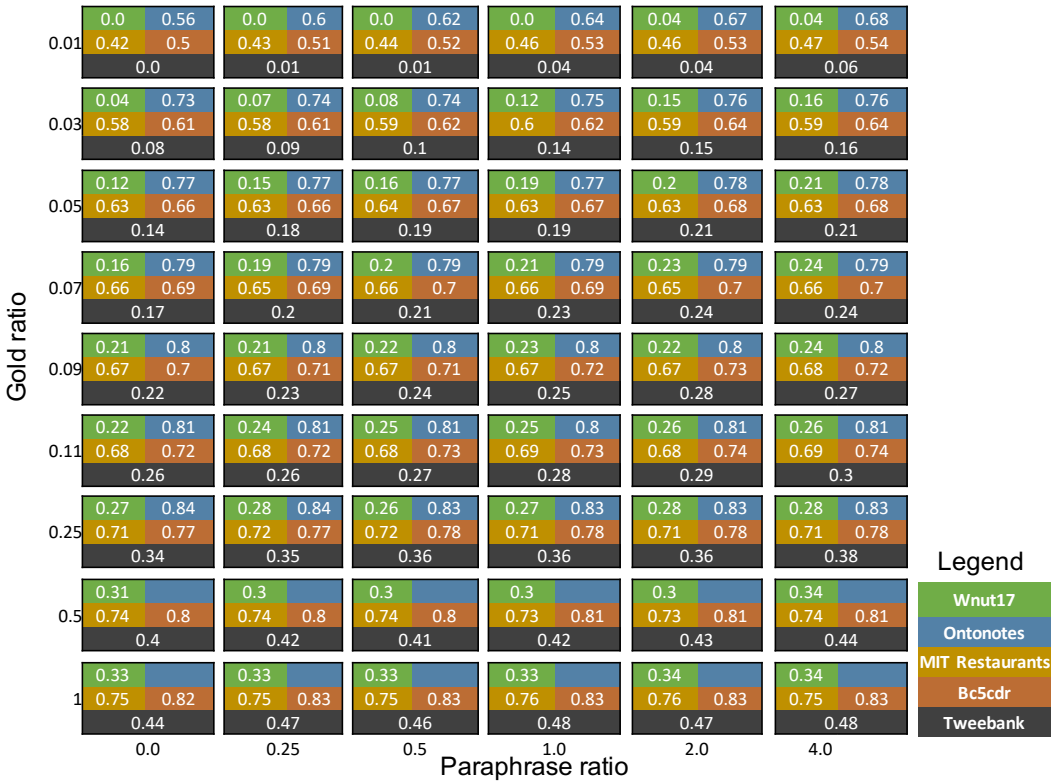


Figure 2: Micro F1 for Davinci (Prompt B) on datasets across gold and paraphrase ratios.

factor	$\hat{\beta}$	t	p
P	0.0148	9.004	1e-15
G	-0.0048	-1.393	0.164
DaV-B	0.0106	3.986	1e-15
support	-0.0031	-5.815	1e-15
cap	-0.0011	-0.473	0.636
number	-0.0154	-5.031	1e-15
P:G	-0.0032	-1.937	0.053
P:DaV-B	0.0078	6.121	1e-15
P:support	-0.0028	-10.925	1e-15
P:cap	0.0008	0.715	0.475
P:number	-0.0165	-11.095	1e-15

Table 4: Coefficients of the model with entity level analysis (See the Appendix A.8 for the coefficients of the full model) $\Delta_{F1} \sim \mathbf{P}araphrase * (\mathbf{G}old + model + support + \mathbf{cap}italize + number)$.

strictly outperform those introduced using heuristics. DaV-B also achieves or matches best performance at G=1 (0.25 for Ontonotes) and P=4 across all datasets (See Appendix A.7). Ada variants show the most inconsistent results, with Backtranslation and Pegasus outperforming them as well as DaV-A in many cases. Full results are available in Appendix A.6.

Similarly, the statistical model (Table 4) shows that both main factor of DaV-B ($\hat{\beta}$ = 0.0106 , p < 1e-15) and its interaction with paraphrase ratio ($\hat{\beta}$ = 0.0078, p < 1e-15) are positive and significant, indicating that as P increases DaV-B has significantly more improvement than the reference model (Ada-A) but other paraphrases do not show such a pattern as the main factors are all insignificant and interactions are inconsistent.

Effect of P and G While we run similar experiments on all paraphraser-dataset pairs, we share the aggregate F1 performance across all G&P configurations of DaV-B on all datasets in Figure 2 (Full results Appendix A.6): We see consistent benefits of paraphrasing at lower gold ratios, and diminishing returns in relative performance bump as we go to higher values. Other paraphrasers show similar trends at low G ratios with some exceptions (Ada variants in BC5CDR, and Backtranslation on MIT-R) (See Figure 1, Appendix A.6), although we see a lot more mixed results at medium to high G ratios.

Our statistical model (see Table 4) reveals similar conclusions: we see the main factor of P ($\hat{\beta}$ = 0.0148, p < 1e-15) is significant and its interaction with G ($\hat{\beta}$ = -0.0032, p < 0.053) are marginally significant. This indicates that P is generally positive correlated with performance gain and there is a weak tendency that the coefficients of P reduces as G increases. In other words, paraphrasing improves the downstream performance but becomes less effective when adding more gold data (a similar trend is also seen in Figure 1).

Effect of Entity Characteristics In terms of entity support, the model shows a significant negative main factor ($\hat{\beta}$ = -0.0031, p < 1e-15) and an interaction with a paraphrase ratio ($\hat{\beta}$ = -0.0028, p < 1e-15), which reveals that the effect of entity support on performance improvement varies based on P with the relationship: improvement $\sim$ constant + (-0.0031 + (-0.0028) $*$ paraphrase)) $*$ support. The negative coefficient of support indicates entity classes with less support are more likely to benefit from an increase of paraphrasing than those with more support.

As for the surface form characteristics, the model reveals a negative interaction ($\hat{\beta}$ = -0.0165, p < 1e-15) and negative main factor ($\hat{\beta}$ = -0.0154, p < 1e-15) for the number form, suggesting the proportion of mentions being a number is negatively correlated with performance improvement and the negative correlation is enhanced as the paraphrasing ratio grows. By contrast, neither the main factor for capitalization nor the interaction is significant, indicating the effect of surface form of capitalization does not play an important role .

3.2 Using only paraphrases for training NER

3.2.1 Experimental setup

We further evaluate quality of paraphrases directly by using **only** synthetic data to train NER models. These experiments are done at P=1 for paraphrases generated from the entire training set (G=1).

3.2.2 Results

Aggregate F1 scores of all paraphrasers are shown in Table 3. We find GPT-3 DaV-B paraphrases performing best across all datasets. The trends among paraphrasers track augmentation performance observed in Figure 1 and Appendix A.6.

3.3 Entity Memorization

Our proposed augmentation and re-annotation strategies in Section 2.2 promote duplication of entity mentions for paraphrases from all paraphrasers. This can lead to shortcut learning (Geirhos et al.,

2020) where the model may just memorize mentions, as opposed to learning features that generalize to unseen mentions (Augenstein et al., 2017). This effect may be observed as a drop in performance in the subsets of our test sets that contain mentions not seen during model training (i.e. an unseen entity test set). We therefore, extend our entity level analysis to also study memorization per entity type, with entity-level *harder* unseen entity (UE) test sets. While a change in performance on UE test sets may come from a combination of factors, we treat a drop to be indicative of memorization. Similarly an increase in performance in UE test set performance with increasing paraphrasing, may indicate a paraphraser that does not promote memorization, but instead improves generalization in the NER model.

3.3.1 Creation of UE test sets

For every entity type in each dataset, we generate UE test sets for all G / P ratio combinations. For any given configuration of G, P, and entity type, an UE test set would include test set samples that contain mentions of that entity not seen within training data for that configuration.

3.3.2 Experimental setup

Models trained for each dataset, G / P ratio, and paraphraser combination in Section 3.1 are evaluated on their respective entity level UE test sets to generate F1 scores per entity type.

To measure the proclivity of paraphrasers to generate synthetic data that promotes memorization we conduct a regression analysis similar to section 3.1.2.

We define memorization as the drop in F1 performance on the UE test sets when paraphrases are added during training at a given G ratio. More formally, the memorization value for given entity at a particular G, P combination is

$$Memorization(g, p, ent) = -\Delta UE_{F1}(g, p, ent)$$

$$= UE_{F1}(g, 0, ent) - UE_{F1}(g, p, ent)$$

3.3.3 Results

Effect of Paraphrasers Based on the statistical model Table 5, DaV-B shows a consistent reduction in memorization on average across all entities ($\hat{\beta}$ = -0.0138) and as P is increased memorization further reduces ($\hat{\beta}$ = -0.0113) suggesting that DaV-B is less susceptible to inducing memorization characteristics in the downstream NER model than the Ada-A model as reference. On the contrary, we see worsening of memorization with most other paraphrasers (Ada-B, BT) on average with Ada-A as reference, while Pegasus at high P does seem to reduce memorization although not to the same extent as Dav-B Appendix A.9. All other interactions with memorization do not pass the 5% significance threshold.

factor	$\hat{\beta}$	t	p
P	-0.01	-5.769	1e-15
G	0.0012	0.323	0.746
DaV-B	-0.0138	-4.937	1e-15
support	0.0041	7.257	1e-15
cap	-0.0018	-0.759	0.448
number	0.0162	5.028	1e-15
P:G	0.0045	2.543	0.011
P:DaV-B	-0.0113	-8.354	1e-15
P:support	0.0031	11.456	1e-15
P:cap	-0.0044	-3.874	1e-15
P:number	0.0142	9.062	1e-15

Table 5: Coefficients of the Linear model for Memorization: memorization $\sim$ **P**araphrase $*$ (**G**old + model + support + **cap**italize + number). Full results Appendix A.9.

Effects of P and G Table 5 suggests that paraphrasing reduces memorization ($\hat{\beta}$ = -0.01) on average across all entities, however, at higher Gs, paraphrasing worsens memorization ($\hat{\beta}$ = 0.0045). Level of G by itself does not significantly interact with memorization (p=0.746>0.05).

Effect of Entity Characteristics Numerical and high support entities seem to have a significant positive interaction with memorization ($\hat{\beta}$ = 0.01162 and 0.0041 respectively) which increases in effect as P is increased ($\hat{\beta}$ = 0.0142 and 0.0031). This implies paraphrasing for entities with a high support generally worsens the performance on unseen entities, indicating memorization. Also, numerical entities seem to be easier for the NER model to memorize. Finally, Capitalized entities at high P has a negative correlation with memorization ($\hat{\beta}$ = -0.0044).

3.3.4 Addressing memorization with mention replacement

We extend our experiments for GPT-3 DaV-B by also incorporating entity mention replacement (MR) into the paraphrasing pipeline. In our approach, we utilize the ability of language models to

be a knowledge base (Petroni et al., 2019), and follow instructions, to source replacement mentions for various entity mentions and types in our training set. In particular, for every entity mention in the gold set, we prompt GPT-3 DaVinci model to generate entity mentions that are similar to the gold entity mention, while also providing a phrase level definition of the entity type being replaced.

Prompt used for mention replacement:

"

Please list 10 examples of `ENTITY_TYPE` *such as* `'ENTITY_VALUE'`*:*

1.

"

ENTITY_VALUE is replaced with the actual gold mention, and ENTITY_TYPE is replaced by a nominal phrase description of the entity class (See Appendix A.1.1 Figure 4). This label conditioned prompt allows us to generate mention replacements closer to the gold entity value, that are more likely to remain consistent with entity label. We use a temperature of 0.8, and a maximum length of 250, with other parameters set to default in the generation. Since our base paraphrases are biased towards entity value retention, we are able to retain span annotation when replacing the entity value in any given paraphrase with an equivalent entity sampled from GPT-3 DaVinci generations. These paraphrases are used as augmentation data to run experiments similar to Section 3.1.

Results Table 6 compares DaV-B to DaV-B MR across all datasets based on relative improvement in overall F1 over no paraphrasing baseline for different G values. Here we see mention replacement especially useful for MIT-R, Tweebank, and WNUT-17 datasets, while being harmful in Ontonotes. Mention replacement makes no significant difference in BC5CDR Appendix A.15.

We also compare the performance of DaV-B vs Dav-B MR for indications of entity memorization. To do so, we introduce a "swapped" feature and conduct statistical analysis similar to Section 3.3.2. Table 7 shows that mention replacement is a good solution to reduce memorization ($\hat{\beta}$ = -0.0065) in general. The interaction between mention replacement as paraphrasing increases is insignificant (p=0.533>0.05) which implies that the coefficient of MR does not vary much as P increases.

	MIT-R	Onto -notes	BC5 -CDR	Twee -bank	Wnut -17
DaV-B	15	**35**	28	14	15
DaV-B +MR	**30**	0	17	**31**	**30**

Table 6: Counts the configurations of G & P where a paraphraser shows highest relative improvement over no paraphrasing baseline for a given G (Division by zero is avoided by using absolute improvement). MR refers to mention replacement. We also conducted Wilcoxon signed-rank tests to evaluate whether the relative performance improvement before or after MR is significantly different. The tests show a significant improvement for MIT-R, Tweebank and Wnut17 and no significant difference for BC5CDR and a significant reduction for Ontonotes.

factor	$\hat{\beta}$	t	p
P	-0.0054	-5.187	0.000
MR	-0.0065	-2.115	0.035
P:MR	0.0009	0.623	0.533

Table 7: Coefficients of the Linear model for Memorization with Mention Replacement: memorization $\sim$ **P**araphrase $*$ **MR**.

4 Future work

While our work proposes a paraphrasing pipeline that performs consistently better than established paraphrasing pipelines in NER, we expect further benefits to come from more exhaustive tuning of prompts used to generate paraphrases. Another potential direction to improve downstream performance is to explore better (than random) sampling strategy for paraphrases (based on entity density, entity recall, or other metrics).

5 Conclusion

We study the effect of six paraphrasing systems on downstream NER performance across 5 datasets. We find that the choice of paraphraser system (model + prompt) strongly affects NER performance. GPT-3 DaV-B performs the best at generating paraphrases capable of improving NER performance while other paraphrasers show mixed results. We further find that generating inline annotations using GPT-3 Davinci works superior to strictly heuristic based annotations. While we find paraphrasing to be more effective at lower amount of training data, it helps at higher levels depending on dataset, and paraphraser. Additionally, we find GPT-3 DaV-B to be most immune against entity mention memorization, with the memorization re-

ducing further with GPT-3 based mention replacement on certain datasets. Our findings speak to the exceptional effectiveness of GPT-3 DaVinci based systems in generating paraphrases promoting generalization in NER applications, thereby making it the de facto choice for paraphrasing in NER.

6 Limitations and risks

This work utilizes generative models trained on large volumes of data, to generate supplemental training data for named entity recognition systems. We do not address any biases, or filter generations of the underlying paraphrasers when using their generated data. This can bias the fine tuned models towards underlying biases of the generative system.

While we do not test or correct the paraphrasing systems for biases, we do not find any evidence for the models deviating unfairly from the underlying training data in any of our human evaluations of the paraphrases.

We recommend human review, and automatic filtering of the generations when applying techniques based on generative models to critical applications, to ensure the black box paraphrasing does not introduce, or exacerbate the biases in existing training datasets.

References

Isabelle Augenstein, Leon Derczynski, and Kalina Bontcheva. 2017. Generalisation in named entity recognition: A quantitative analysis. *Computer Speech & Language*, 44:61–83.

Ondřej Bojar and Aleš Tamchyna. 2011. Improving translation model by monolingual data. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 330–336, Edinburgh, Scotland. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Jon Ander Campos and Jun Shern. 2022. Training language models with language feedback. In *ACL Workshop on Learning with Natural Language Supervision. 2022.*

Jean-Philippe Corbeil and Hadi Abdi Ghadivel. 2020. Bet: A backtranslation approach for easy data augmentation in transformer-based paraphrase identification context. *arXiv preprint arXiv:2009.12452*.

Xiang Dai and Heike Adel. 2020. An analysis of simple data augmentation for named entity recognition. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3861–3867, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the WNUT2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147, Copenhagen, Denmark. Association for Computational Linguistics.

Bosheng Ding, Linlin Liu, Lidong Bing, Canasai Kruengkrai, Thien Hai Nguyen, Shafiq Joty, Luo Si, and Chunyan Miao. 2020. DAGA: Data augmentation with a generation approach for low-resource tagging tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6045–6057, Online. Association for Computational Linguistics.

Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 968–988, Online. Association for Computational Linguistics.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. 2020. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.

Charles R Harris, K Jarrod Millman, Stéfan J Van Der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J Smith, et al. 2020. Array programming with numpy. *Nature*, 585(7825):357–362.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. OntoNotes: The 90% solution. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 57–60, New York City, USA. Association for Computational Linguistics.

Hang Jiang, Yining Hua, Doug Beeferman, and Deb Roy. 2022. Annotating the tweebank corpus on named entity recognition and building NLP models for social media analysis. *CoRR*, abs/2201.07281.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, et al. 2021. Datasets: A community library for natural language processing. *arXiv preprint arXiv:2109.02846*.

Kun Li, Chengbo Chen, Xiaojun Quan, Qing Ling, and Yan Song. 2020. Conditional augmentation for aspect term extraction via masked sequence-to-sequence generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7056–7066, Online. Association for Computational Linguistics.

Jian Liu, Yufeng Chen, and Jinan Xu. 2022. Low-resource ner by data augmentation with prompting. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 4252–4258. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Jingjing Liu, Panupong Pasupat, Yining Wang, Scott Cyphers, and Jim Glass. 2013. Query understanding enhanced by hierarchical parsing structures. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 72–77. IEEE.

Wes McKinney et al. 2011. pandas: a foundational python library for data analysis and statistics. *Python for high performance and scientific computing*, 14(9):1–9.

Tong Niu, Semih Yavuz, Yingbo Zhou, Nitish Shirish Keskar, Huan Wang, and Caiming Xiong. 2020. Unsupervised paraphrasing with pretrained language models. *arXiv preprint arXiv:2010.12885*.

Itsuki Okimura, Machel Reid, Makoto Kawano, and Yutaka Matsuo. 2022. On the impact of data augmentation on downstream performance in natural language processing. In *Proceedings of the Third Workshop on Insights from Negative Results in NLP*, pages 88–93.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*.

Amane Sugiyama and Naoki Yoshinaga. 2019. Data augmentation using back-translation for context-aware neural machine translation. In *Proceedings of the Fourth Workshop on Discourse in Machine Translation (DiscoMT 2019)*, pages 35–44, Hong Kong, China. Association for Computational Linguistics.

Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT — Building open translation services for the World. In *Proceedings of the 22nd Annual Conferenec of the European Association for Machine Translation (EAMT)*, Lisbon, Portugal.

Asahi Ushio and Jose Camacho-Collados. 2021. T-NER: An all-round python library for transformer-based named entity recognition. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 53–62, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. 2020. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature methods*, 17(3):261–272.

Chih-Hsuan Wei, Yifan Peng, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Jiao Li, Thomas C Wiegers, and Zhiyong Lu. 2016. Assessing the state of the art in biomedical relation extraction: overview of the biocreative v chemical-disease relation (cdr) task. *Database*, 2016.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Zhu Wenjing, Liu Jian, Xu Jinan, Chen Yufeng, and Zhang Yujie. 2021. Improving low-resource named entity recognition via label-aware data augmentation and curriculum denoising. In *Proceedings of the 20th Chinese National Conference on Computational Linguistics*, pages 1131–1142, Huhhot, China. Chinese Information Processing Society of China.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. *Advances in Neural Information Processing Systems*, 33:6256–6268.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

Ran Zhou, Xin Li, Ruidan He, Lidong Bing, Erik Cambria, Luo Si, and Chunyan Miao. 2022. MELM: Data augmentation with masked entity language modeling for low-resource NER. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2251–2262, Dublin, Ireland. Association for Computational Linguistics.

A Appendix

The appendix includes prompt design and multiline generation, human annotation guideline, paraphrase generation quality analysis, analysis of the interaction between gold and paraphrase ratio for each dataset, downstream F1 score for each dataset, risks and limitations as well as software acknowledgements.

A.1 Prompt design

A.1.1 Entity mention replacement prompts

The following prompt is used in the entity mention replacement pipeline to generate entity values similar to gold mentions: ENTITY_TYPE is replaced by a phrase that explains the entity in a few words using

```
Please list 10 examples of {ENTITY_TYPE} such as '{ENTITY_VALUE}':
1.
```

Figure 3: GPT-3 DaVinci is instructed to generate mention replacements for ENTITY_VALUE of the type ENTITY_TYPE.

the following table: Here is an example for the prompt used for entity mention replacement along with

Dataset	Entity type	Replacement
tner_bc5cdr	CHEMICAL	chemical
tner_bc5cdr	DISEASE	disease
tner_mit_restaurant	PRICE	price
tner_mit_restaurant	CUISINE	cuisine
tner_mit_restaurant	LOCATION	location
tner_mit_restaurant	RESTAURANT_NAME	restaurant name
tner_mit_restaurant	AMENITY	amenity
tner_mit_restaurant	RATING	rating
tner_mit_restaurant	HOURS	hours
tner_mit_restaurant	DISH	dish
tner_ontonotes5	CARDINAL	cardinal value
tner_ontonotes5	DATE	date
tner_ontonotes5	PERSON	person
tner_ontonotes5	NORP	nationalities or religious or political groups
tner_ontonotes5	GPE	countries or cities or states
tner_ontonotes5	LAW	named documents made into laws
tner_ontonotes5	ORG	companies or agencies or institutions
tner_ontonotes5	PERCENT	percentage
tner_ontonotes5	ORDINAL	ordinal value
tner_ontonotes5	MONEY	money
tner_ontonotes5	WORK_OF_ART	work of art (titles of books, songs, etc)
tner_ontonotes5	FAC	facilities (buildings, airports, highways, bridges, etc)
tner_ontonotes5	TIME	time smaller than a day
tner_ontonotes5	LOC	location (mountains, ranges, bodies of water)
tner_ontonotes5	QUANTITY	quantity measurement for weight or distance
tner_ontonotes5	PRODUCT	product (vehicles, weapons, foods, etc)
tner_ontonotes5	EVENT	event (named hurricanes, battles, wars, sports events, etc)
tner_ontonotes5	LANGUAGE	language
tner_tweebank_ner	ORG	organization
tner_tweebank_ner	PER	person
tner_tweebank_ner	LOC	locations
tner_tweebank_ner	MISC	named entities that are not locations, persons, organizations
tner_wnut2017	LOCATION	geopolitical locations and facilities
tner_wnut2017	GROUP	group name
tner_wnut2017	CORPORATION	corporation
tner_wnut2017	PERSON	person
tner_wnut2017	CREATIVE_WORK	creative work (song, movie, book and so on)
tner_wnut2017	PRODUCT	product name (tangible goods, or well-defined services)

Figure 4: ENTITY_TYPE is replaced by replacement phrases for each entity type.

generation from GPT-3 DaVinci:

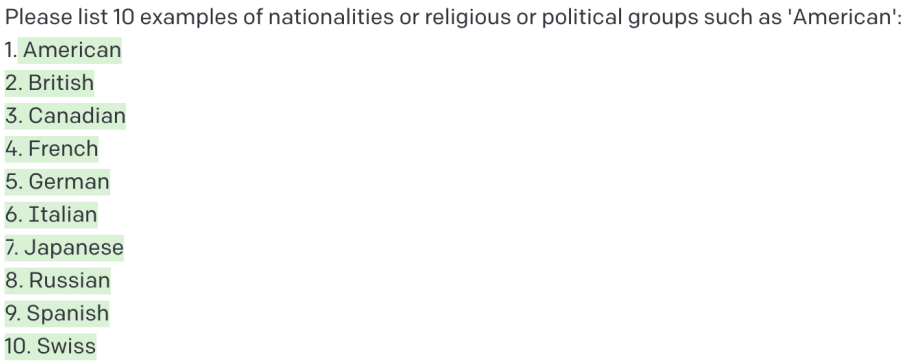


Figure 5: Mention replacement prompt and output from GPT-3 DaVinci.

A.2 Multiline generation

LLM paraphrasers can be triggered to generate multi-line outputs. This behavior is more common in Ada variants over DaVinci, showing the DaVinci is better at following prompt instructions.

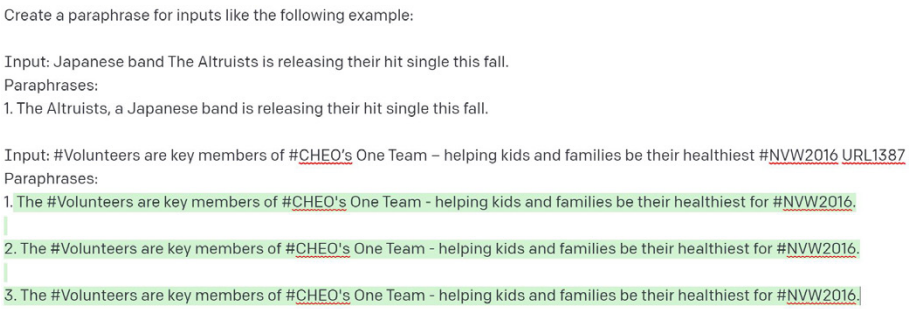


Figure 6: GPT-3 variants sometimes generate multiple numbered paraphrases. We choose to retain only the first paraphrase in these cases.

A.3 Human evaluation guidelines

See Figure 7 for annotation guideline.

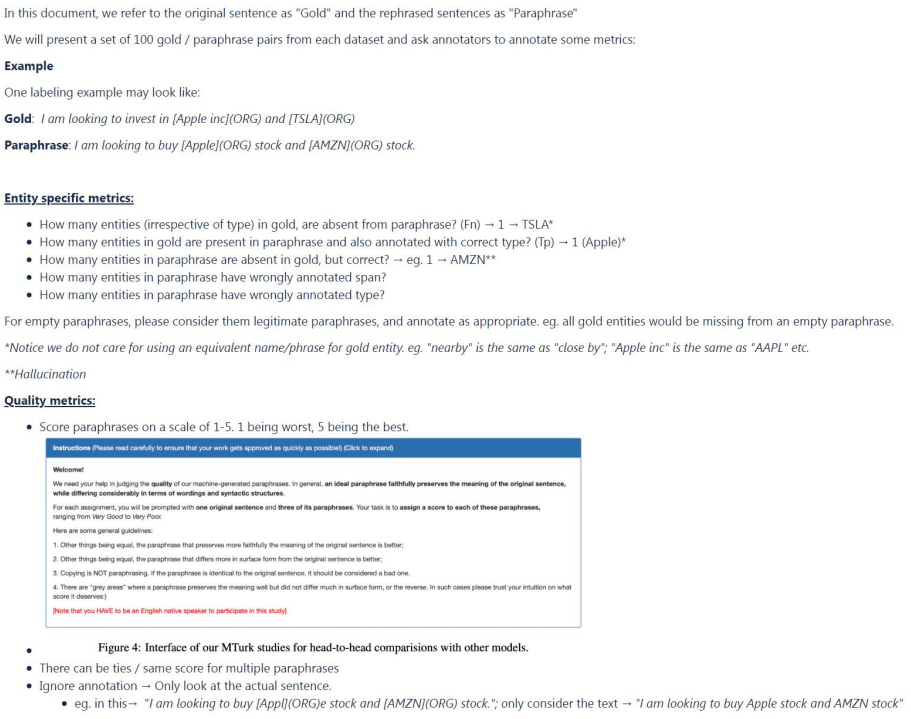
A.4 Paraphrase generation quality Analysis

Besides assessing usefulness for NER with actual training, we investigate paraphrase generation quality directly from two perspectives – entity preservation and paraphrase quality to see to what extent these metrics correlate with NER performance.

As entities are central to NER, we hypothesize entity preservation to be important for performance. We count the number of gold entities that appear in paraphrases with correct annotations via a case insensitive string match (entity recall). This calculation sets a lower bound of the entity preservation accuracy.

Good paraphrases are also expected to introduce form variety while preserving the meaning faithfully, potentially helping downstream performance. We asked three human annotators to annotate paraphrases generated by the six systems for 50 training examples sampled for each dataset. Specifically, human annotators were instructed to ignore the entity accuracy and to score paraphrases from 1-5 based on the paraphrasing quality. Our guidelines are similar to (Niu et al., 2020) (Appendix A.3). The annotator are from the internal data annotator team hired by the company and the annotation task is assigned as the annotation work.

According to Figure 8(a), among all the paraphrase systems Davinci (Prompt B) has the highest entity recall rate, followed by Davinci (Prompt A) and backtranslation. While, Ada and Pegasus are more likely to lose gold entities. This suggests a large-sized GPT-3 model with an appropriate prompt can generate examples with high-quality inline entity annotations but a small-sized GPT-3 consistently underperforms



In this document, we refer to the original sentence as "Gold" and the rephrased sentences as "Paraphrase"

We will present a set of 100 gold / paraphrase pairs from each dataset and ask annotators to annotate some metrics:

Example

One labeling example may look like:

Gold: *I am looking to invest in [Apple inc](ORG) and [TSLA](ORG)*

Paraphrase: *I am looking to buy [Apple](ORG) stock and [AMZN](ORG) stock.*

Entity specific metrics:

- How many entities (irrespective of type) in gold, are absent from paraphrase? (Fn) → 1 → TSLA*
- How many entities in gold are present in paraphrase and also annotated with correct type? (Tp) → 1 (Apple)*
- How many entities in paraphrase are absent in gold, but correct? → eg. 1 → AMZN**
- How many entities in paraphrase have wrongly annotated span?
- How many entities in paraphrase have wrongly annotated type?

For empty paraphrases, please consider them legitimate paraphrases, and annotate as appropriate. eg. all gold entities would be missing from an empty paraphrase.

**Notice we do not care for using an equivalent name/phrase for gold entity. eg. "nearby" is the same as "close by"; "Apple inc" is the same as "AAPL" etc.*

***Hallucination*

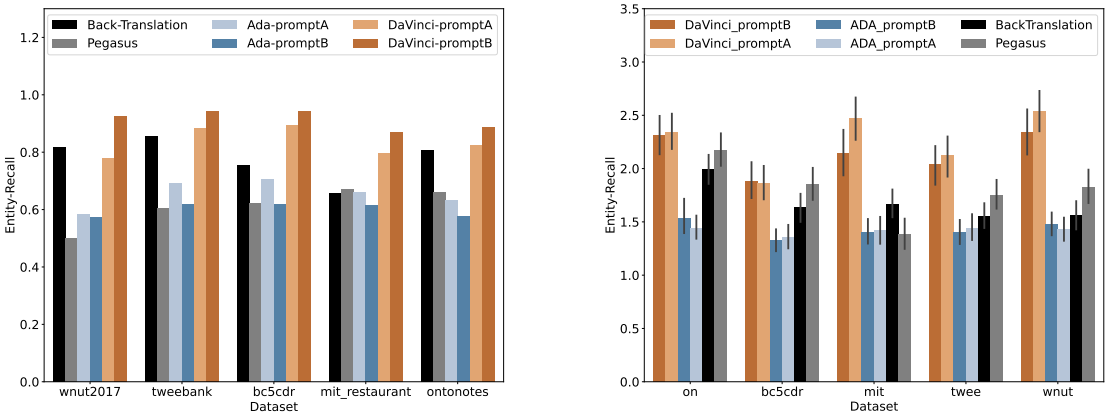
Quality metrics:

- Score paraphrases on a scale of 1-5. 1 being worst, 5 being the best.

Figure 4: Interface of our MTurk studies for head-to-head comparisions with other models.

- There can be ties / same score for multiple paraphrases
- Ignore annotation → Only look at the actual sentence.
  - eg. in this→ *"I am looking to buy [Appl](ORG)e stock and [AMZN](ORG) stock."*; only consider the text → *"I am looking to buy Apple stock and AMZN stock"*

Figure 7: Annotation guideline.



(a) Entity recall evaluation.

(b) Human evaluation of paraphrasing quality.

Figure 8: Paraphrase Evaluation.

even a simple Back-translation system. Figure 8(b) shows Davinci systems always have the best human evaluation scores across datasets followed by Pegasus and Back-translation, while Ada systems are consistently the worst (missing value of Pegasus for mit restaurant is due to technical issue).

In summary, we find that paraphrases generated by the Davinci (Prompt B) system often preserve entities and are of a good paraphrasing quality whereas Ada systems consistently underperform other systems in both metrics across datasets. These results are partially consistent with the downstream evaluations in that the augmentation data generated by Davinci (Prompt B) have reliably better downstream performance compared to other systems. However, broader trends in paraphrasing quality do not track with downstream NER performance.

A.5 Overview of Models Parameters and Downstream Performance

	MIT-R	Ontonotes	BC5CDR	Tweebank	Wnut 17
BT (∼738M)	1	0	2	0	5
Pegasus (∼568M)	1	0	13	3	8
Ada-A (∼350M)	10	0	0	11	0
Ada-B (∼350M)	4	0	0	**16**	2
DaV-A (∼175B)	3	0	4	5	3
DaV-B (∼175B)	**26**	**35**	**26**	10	**27**

Table 8: Counts the configurations of G & P where a paraphraser shows highest relative improvement over no paraphrasing baseline for a given G (Division by zero is avoided by using absolute improvement). GPT-3 DaV-B outperforms other paraphrasers across most datasets. Detailed results of downstream performance for all datasets, paraphrasers, configurations can be found in Appendix A.6

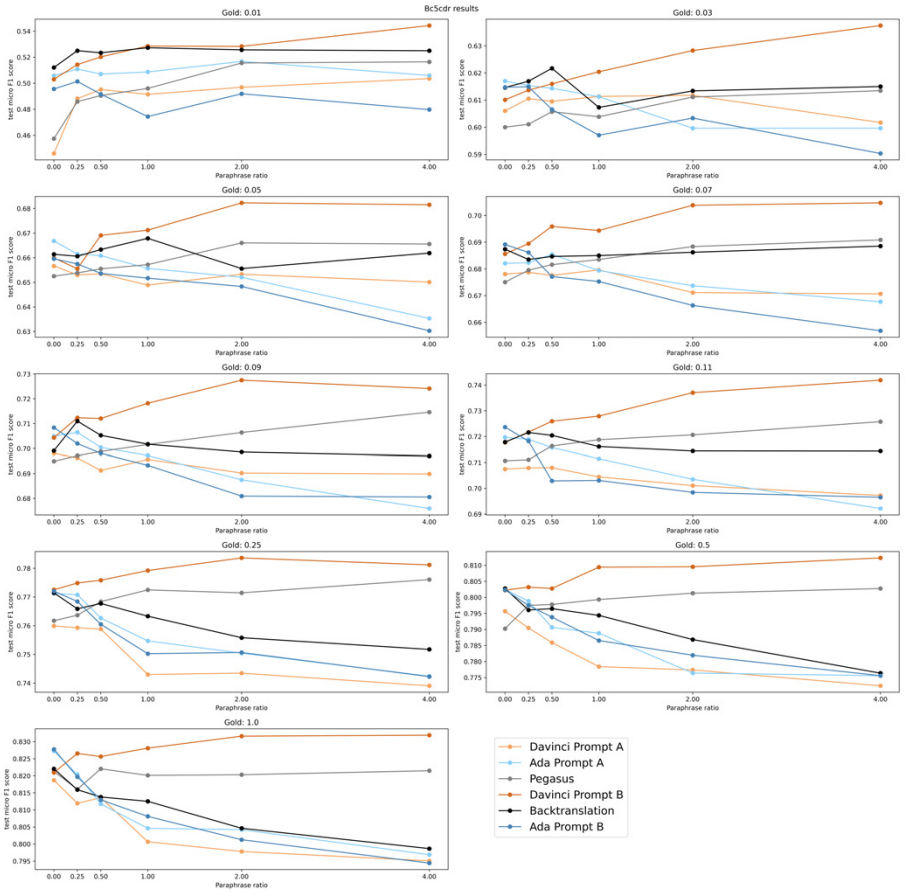
	MIT-R	Onto -notes	BC5 -CDR	Twee -bank	Wnut -17
BT (∼738M)	0.66	0.74	0.76	**0.41**	0.30
Pegasus (∼ 568M)	0.68	0.75	0.78	0.33	0.23
Ada-A (∼350M)	0.71	0.73	0.74	0.36	0.23
Ada-B (∼350M)	0.70	0.72	0.74	0.34	0.23
DaV-A (∼175B)	0.67	0.75	0.76	0.39	0.27
DaV-B (∼175B)	**0.73**	**0.80**	**0.82**	**0.41**	**0.32**

Table 9: Test micro-F1 when training using only paraphrases with P=1 for full dataset. Number in bold is the maximum for a given dataset. GPT-3 DaV-B outperforms all paraphrasers across datasets.

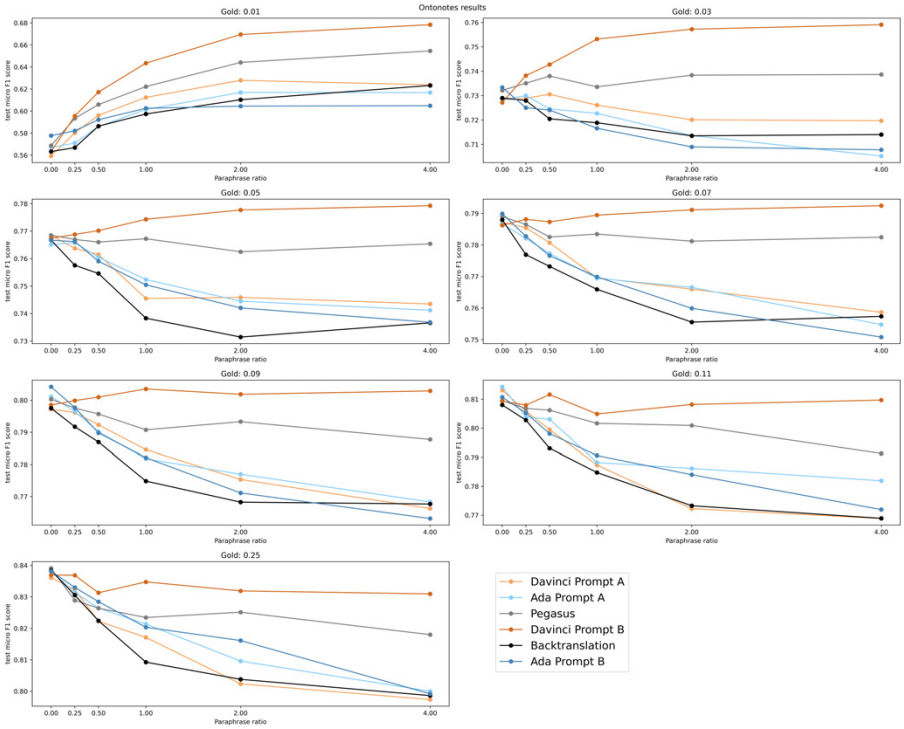
The two tables show the downstream performance aligned with the model size of paraphrasers. We found it that large models (davinci) models only with a reasonable prompt show the advantages over the other smaller models that have much less parameters.

A.6 Detailed results across different gold data sizes for all datasets

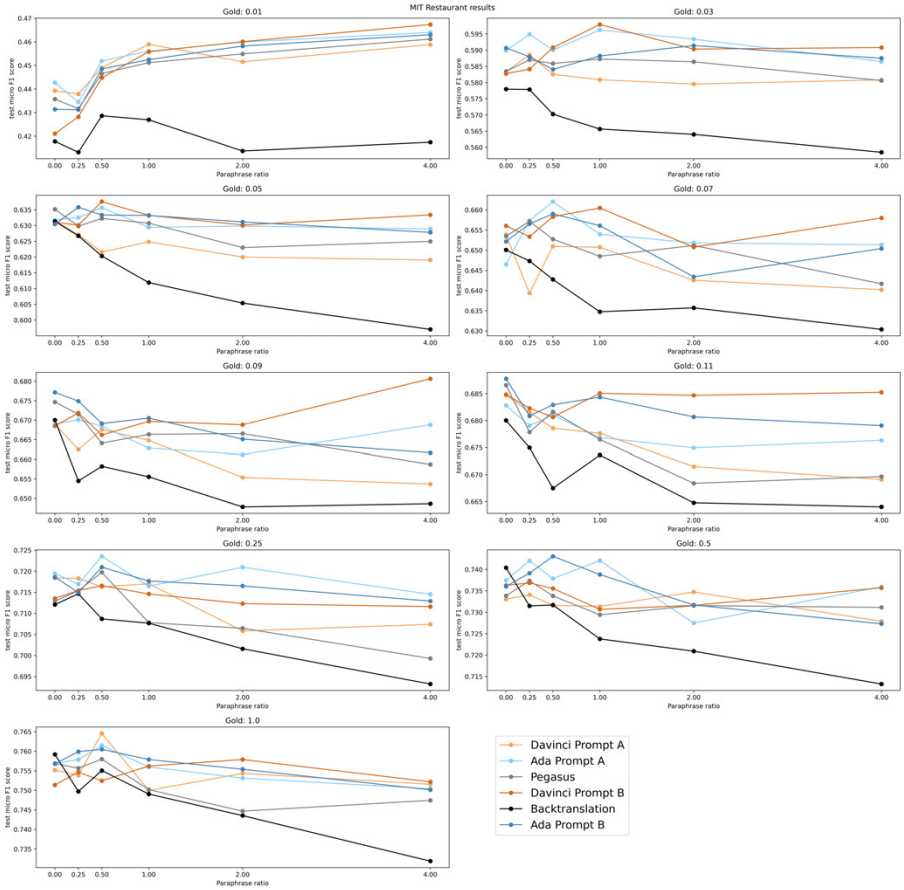
A.6.1 BC5CDR



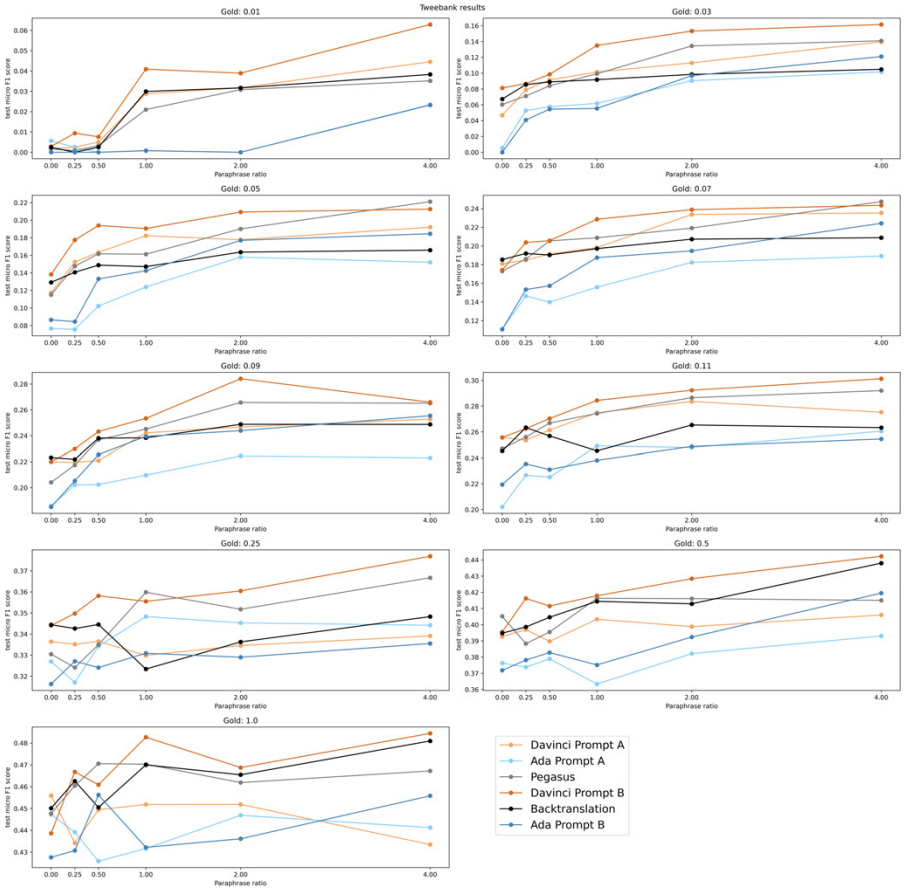
A.6.2 Ontonotes



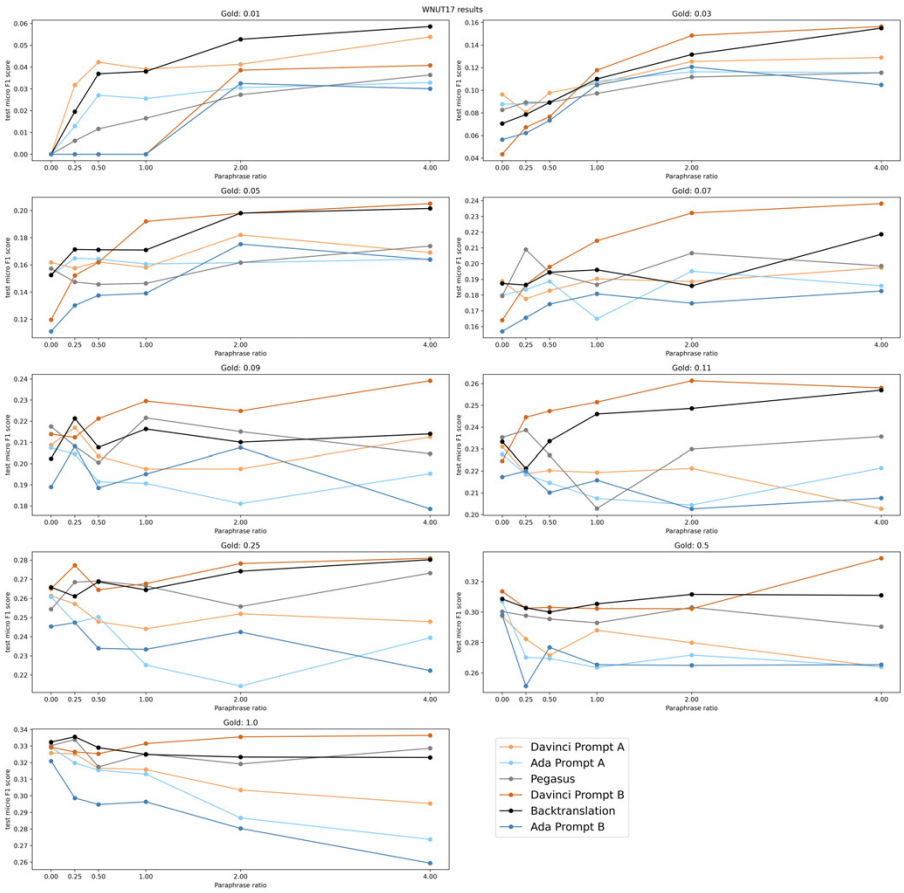
A.6.3 MIT-R


MIT Restaurant results
Gold: 0.01
Gold: 0.03
Gold: 0.05
Gold: 0.07
Gold: 0.09
Gold: 0.11
Gold: 0.25
Gold: 0.5
Gold: 1.0
test micro F1 score
Paraphrase ratio
Davinci Prompt A
Ada Prompt A
Pegasus
Davinci Prompt B
Backtranslation
Ada Prompt B

A.6.4 Tweebank

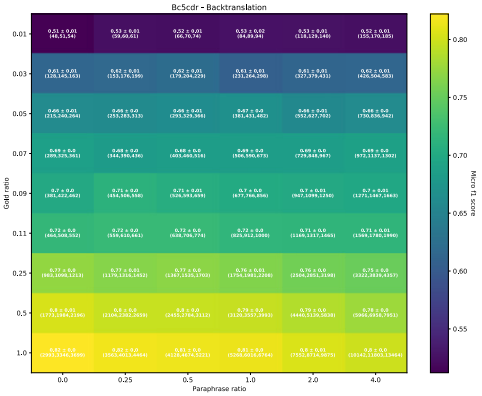


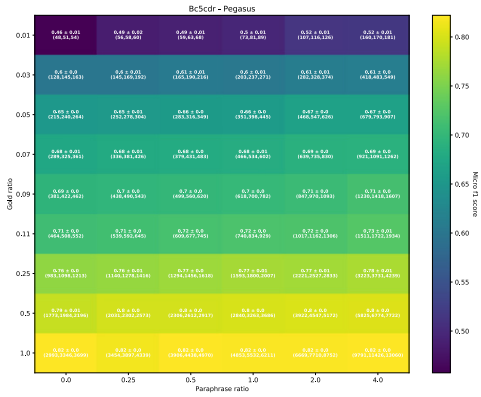
A.6.5 WNUT-17

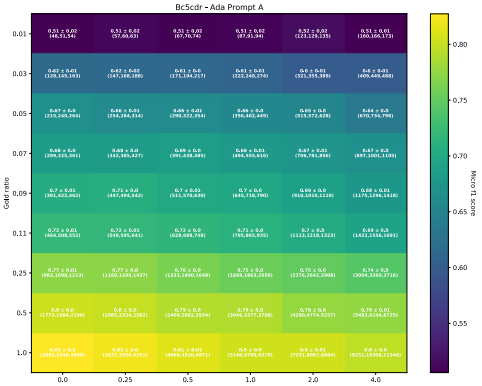

WNUT17 results
Gold: 0.01
Gold: 0.03
Gold: 0.05
Gold: 0.07
Gold: 0.09
Gold: 0.11
Gold: 0.25
Gold: 0.5
Gold: 1.0
test micro F1 score
Paraphrase ratio
Davinci Prompt A
Ada Prompt A
Pegasus
Davinci Prompt B
Backtranslation
Ada Prompt B

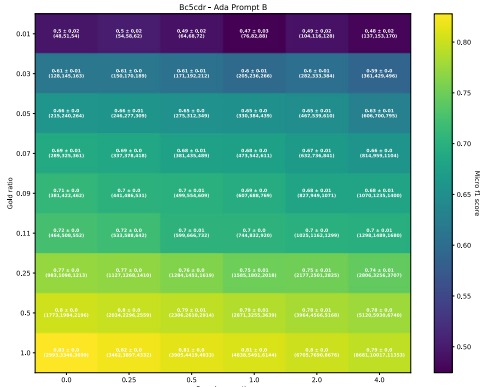
A.7 Heatmap of micro-F1 scores across all datasets & paraphrasers

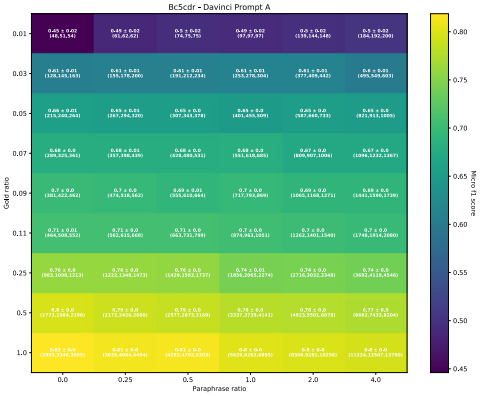
A.7.1 BC5CDR

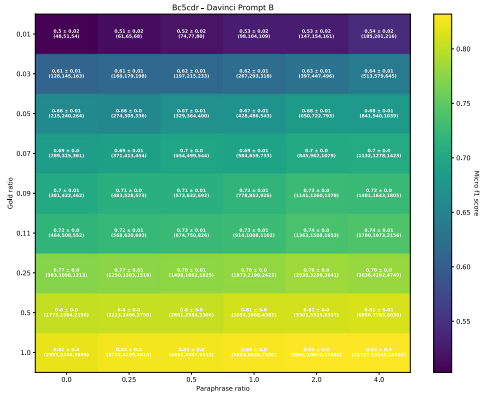




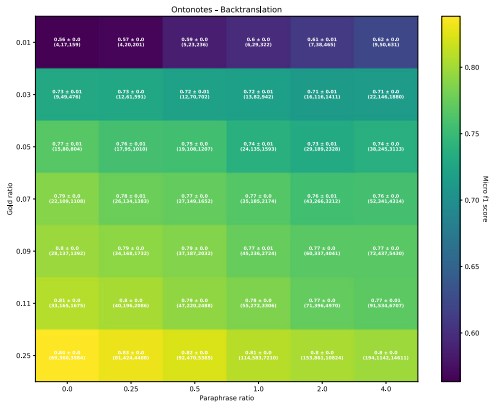


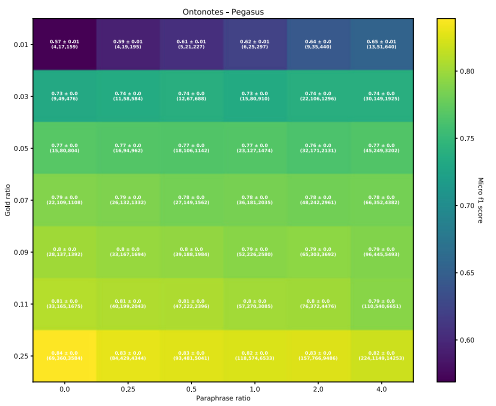


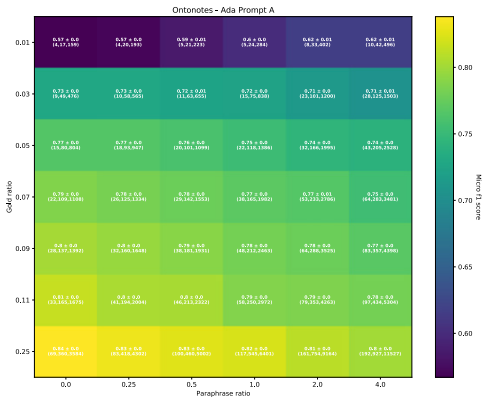


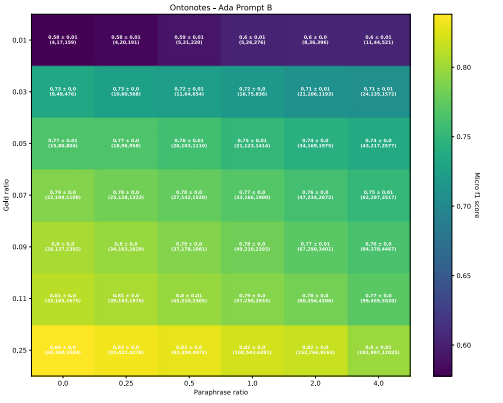


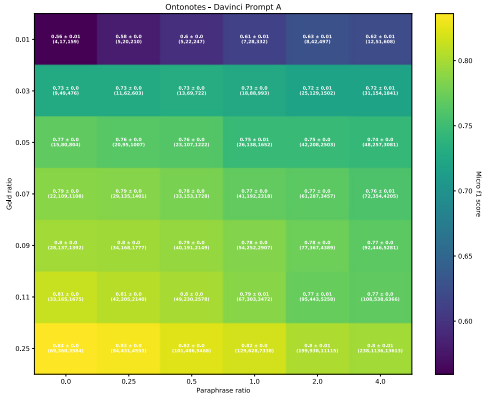
A.7.2 Ontonotes

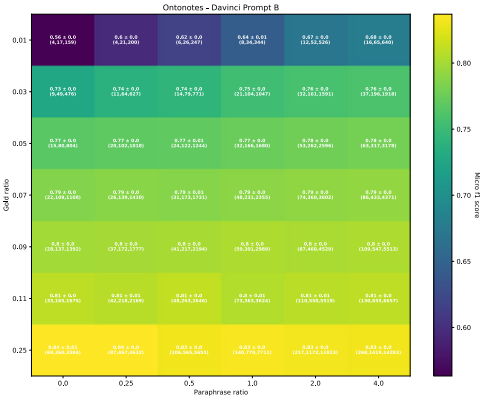




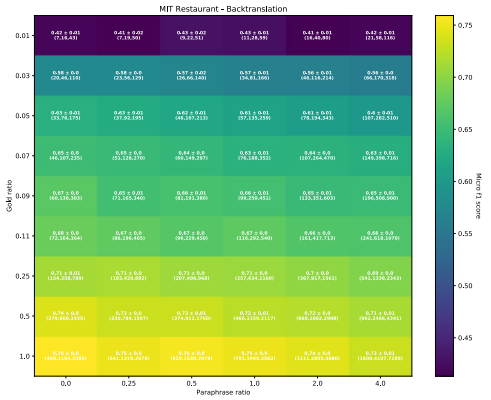


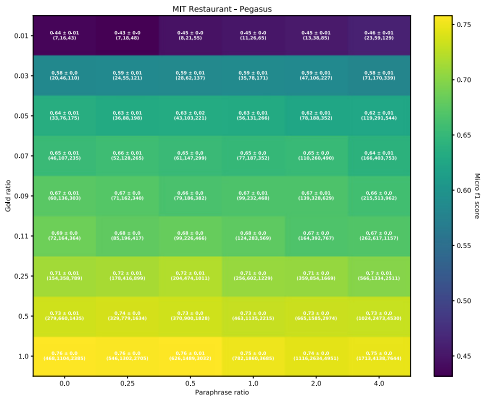


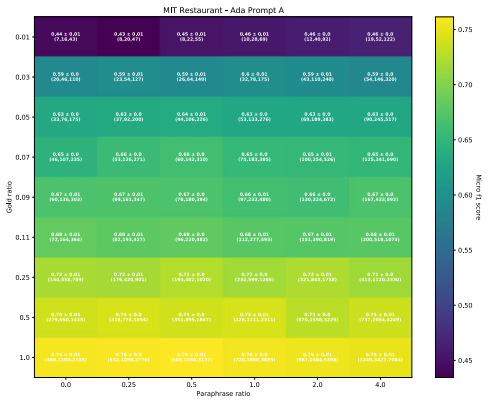


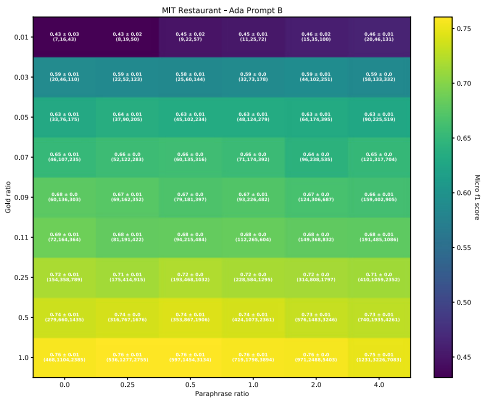


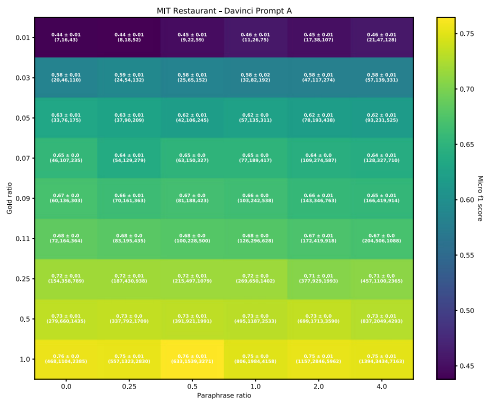
A.7.3 MIT-R

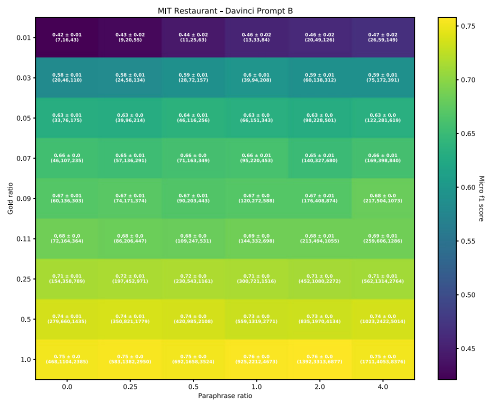

MIT Restaurant - Backtranslation
GoM ratio
Paraphrase ratio
Micro f1 score


MIT Restaurant - Pegasus
GoM ratio
Paraphrase ratio
Micro f1 score

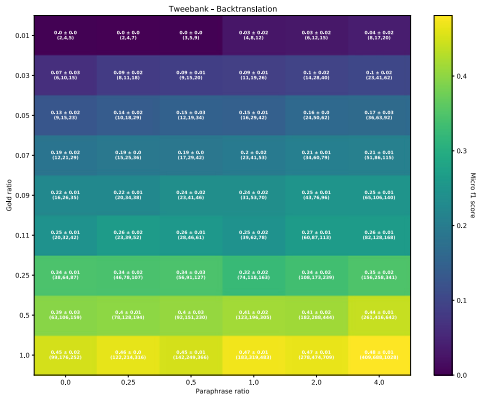

MIT Restaurant - Ada Prompt A
GoM ratio
Paraphrase ratio
Micro f1 score

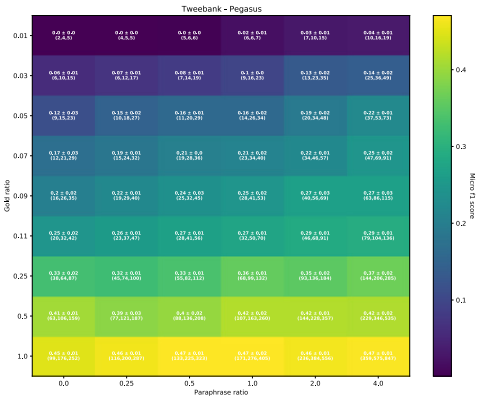

MIT Restaurant - Ada Prompt B
GoM ratio
Paraphrase ratio
Micro f1 score

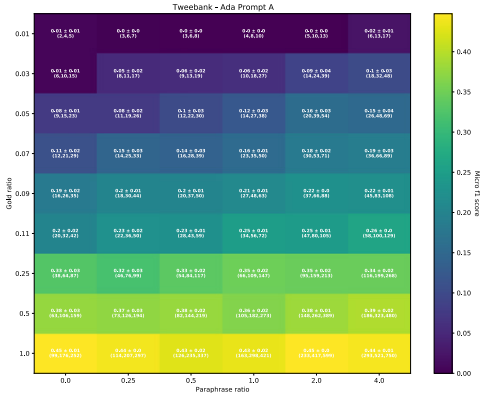

MIT Restaurant - Davinci Prompt A
GoM ratio
Paraphrase ratio
Micro f1 score

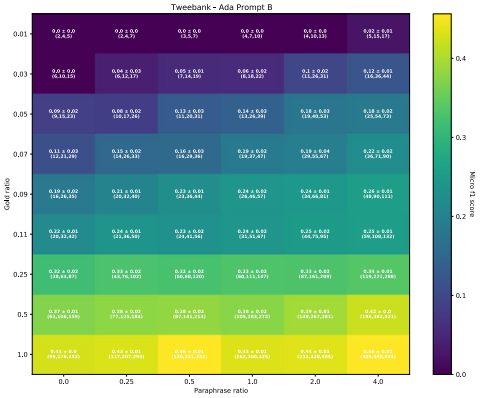

MIT Restaurant - Davinci Prompt B
GoM ratio
Paraphrase ratio
Micro f1 score

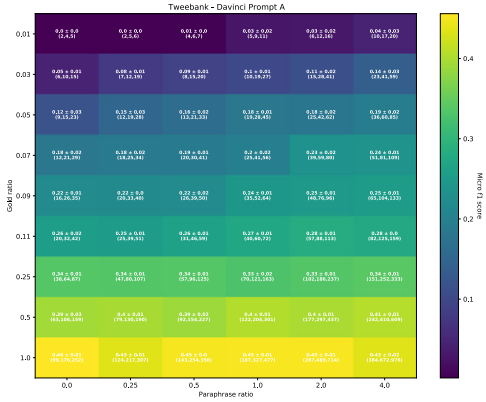
A.7.4 Tweebank

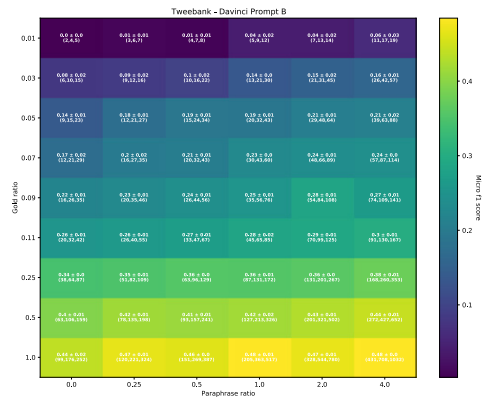




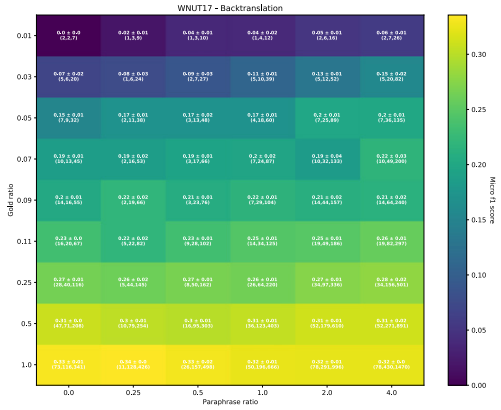


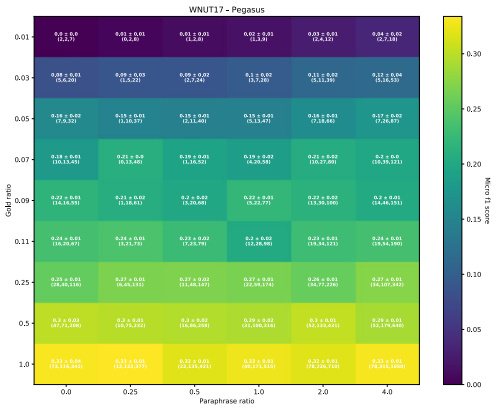


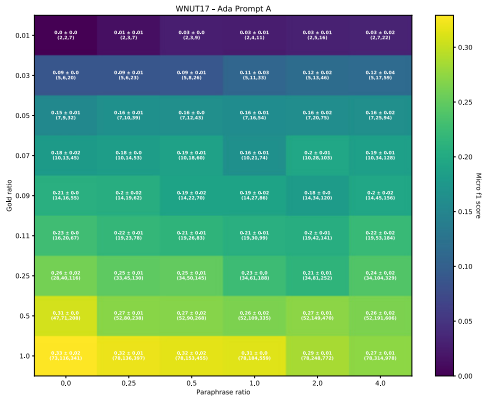


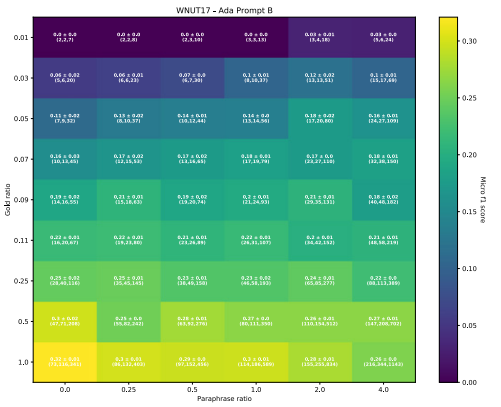


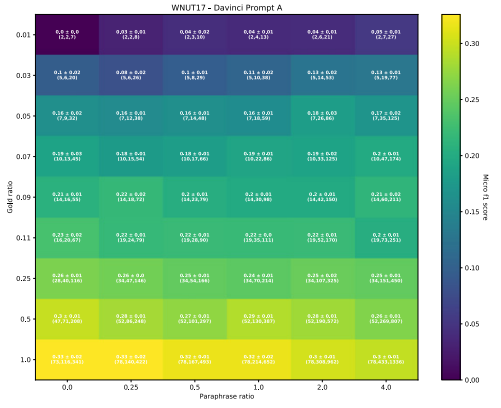
A.7.5 WNUT-17

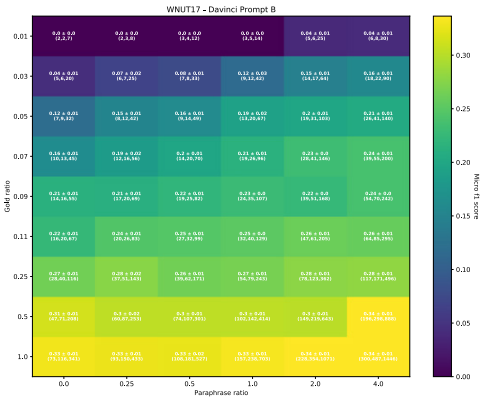

WNUT17 - Backtranslation
Gold ratio
Paraphrase ratio
Micro f1 score


WNUT17 - Pegasus
Gold ratio
Paraphrase ratio
Micro f1 score


WNUT17 - Ada Prompt A
Gold ratio
Paraphrase ratio
Micro f1 score


WNUT17 - Ada Prompt B
Gold ratio
Paraphrase ratio
Micro f1 score


WNUT17 - Davinci Prompt A
Gold ratio
Paraphrase ratio
Micro f1 score


WNUT17 - Davinci Prompt B
Gold ratio
Paraphrase ratio
Micro f1 score

A.8 Entity Level Analysis

OLS Regression Results

Dep. Variable:	change	R-squared:	0.188
Model:	OLS	Adj. R-squared:	0.187
Method:	Least Squares	F-statistic:	111.9
Date:	Sat, 08 Oct 2022	Prob (F-statistic):	0.00
Time:	15:20:34	Log-Likelihood:	14727.
No. Observations:	9180	AIC:	-2.941e+04
Df Residuals:	9160	BIC:	-2.927e+04
Df Model:	19		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	0.0187	0.003	5.505	0.000	0.012	0.025
model[T.ada_promptB]	-0.0013	0.003	-0.499	0.618	-0.007	0.004
model[T.bt]	-0.0048	0.003	-1.786	0.074	-0.010	0.000
model[T.davinci_promptA]	-0.0010	0.003	-0.390	0.696	-0.006	0.004
model[T.davinci_promptB]	0.0106	0.003	3.986	0.000	0.005	0.016
model[T.pegasus]	0.0033	0.003	1.247	0.212	-0.002	0.009
p	0.0148	0.002	9.004	0.000	0.012	0.018
p:model[T.ada_promptB]	-0.0004	0.001	-0.298	0.766	-0.003	0.002
p:model[T.bt]	1.799e-05	0.001	0.014	0.989	-0.003	0.003
p:model[T.davinci_promptA]	-0.0012	0.001	-0.943	0.346	-0.004	0.001
p:model[T.davinci_promptB]	0.0078	0.001	6.028	0.000	0.005	0.010
p:model[T.pegasus]	0.0044	0.001	3.444	0.001	0.002	0.007
g	-0.0048	0.003	-1.393	0.164	-0.012	0.002
log_support_median_base	-0.0031	0.001	-5.815	0.000	-0.004	-0.002
new_capitalization_median	-0.0011	0.002	-0.473	0.636	-0.005	0.003
new_number_median	-0.0154	0.003	-5.031	0.000	-0.021	-0.009
p:g	-0.0032	0.002	-1.937	0.053	-0.007	3.86e-05
p:log_support_median_base	-0.0028	0.000	-10.925	0.000	-0.003	-0.002
p:new_capitalization_median	0.0008	0.001	0.715	0.475	-0.001	0.003
p:new_number_median	-0.0165	0.001	-11.095	0.000	-0.019	-0.014

Omnibus:	4991.319	Durbin-Watson:	1.836
Prob(Omnibus):	0.000	Jarque-Bera (JB):	492136.167
Skew:	1.690	Prob(JB):	0.00
Kurtosis:	38.710	Cond. No.	116.

Figure 9: Linear regression model for Entity Level Analysis.

A.9 Memorization Analysis

A.9.1 Entity Level Memorization

OLS Regression Results

Dep. Variable:	memorization_delta	R-squared:	0.232
Model:	OLS	Adj. R-squared:	0.230
Method:	Least Squares	F-statistic:	145.6
Date:	Mon, 10 Oct 2022	Prob (F-statistic):	0.00
Time:	09:10:34	Log-Likelihood:	14262.
No. Observations:	9180	AIC:	−2.848e+04
Df Residuals:	9160	BIC:	−2.834e+04
Df Model:	19		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	−0.0155	0.004	−4.350	0.000	−0.023	−0.009
model[T.ada_promptB]	0.0035	0.003	1.254	0.210	−0.002	0.009
model[T.bt]	0.0071	0.003	2.522	0.012	0.002	0.013
model[T.davinci_promptA]	−0.0012	0.003	−0.422	0.673	−0.007	0.004
model[T.davinci_promptB]	−0.0138	0.003	−4.937	0.000	−0.019	−0.008
model[T.pegasus]	−0.0040	0.003	−1.422	0.155	−0.009	0.002
p	−0.0100	0.002	−5.769	0.000	−0.013	−0.007
p:model[T.ada_promptB]	0.0004	0.001	0.307	0.759	−0.002	0.003
p:model[T.bt]	−9.022e−05	0.001	−0.066	0.947	−0.003	0.003
p:model[T.davinci_promptA]	−0.0010	0.001	−0.702	0.483	−0.004	0.002
p:model[T.davinci_promptB]	−0.0113	0.001	−8.354	0.000	−0.014	−0.009
p:model[T.pegasus]	−0.0060	0.001	−4.432	0.000	−0.009	−0.003
new_capitalization_median	−0.0018	0.002	−0.759	0.448	−0.006	0.003
new_number_median	0.0162	0.003	5.028	0.000	0.010	0.023
log_base_median_support	0.0041	0.001	7.257	0.000	0.003	0.005
g	0.0012	0.004	0.323	0.746	−0.006	0.008
p:new_capitalization_median	−0.0044	0.001	−3.874	0.000	−0.007	−0.002
p:new_number_median	0.0142	0.002	9.062	0.000	0.011	0.017
p:log_base_median_support	0.0031	0.000	11.456	0.000	0.003	0.004
p:g	0.0045	0.002	2.543	0.011	0.001	0.008

Omnibus:	1751.645	Durbin-Watson:	1.821
Prob(Omnibus):	0.000	Jarque-Bera (JB):	38026.707
Skew:	0.322	Prob(JB):	0.00
Kurtosis:	12.950	Cond. No.	116.

Figure 10: Linear regression model for Memorization Analysis.

A.9.2 Memorization Mention Replacement

OLS Regression Results

Dep. Variable:	memorization_delta	R-squared:	0.017
Model:	OLS	Adj. R-squared:	0.016
Method:	Least Squares	F-statistic:	17.20
Date:	Sun, 09 Oct 2022	Prob (F-statistic):	4.55e−11
Time:	22:16:58	Log-Likelihood:	4366.5
No. Observations:	2937	AIC:	−8725.
Df Residuals:	2933	BIC:	−8701.
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	−0.0084	0.002	−3.865	0.000	−0.013	−0.004
p	−0.0054	0.001	−5.187	0.000	−0.007	−0.003
swapped	−0.0065	0.003	−2.115	0.035	−0.012	−0.000
p:swapped	0.0009	0.001	0.623	0.533	−0.002	0.004

Omnibus:	651.024	Durbin-Watson:	1.083
Prob(Omnibus):	0.000	Jarque-Bera (JB):	2144.468
Skew:	−1.100	Prob(JB):	0.00
Kurtosis:	6.562	Cond. No.	9.36

Figure 11: Statistical analysis of Memorization with Mention Replacement.

A.10 Wnut17 Unseen Entity Set F1

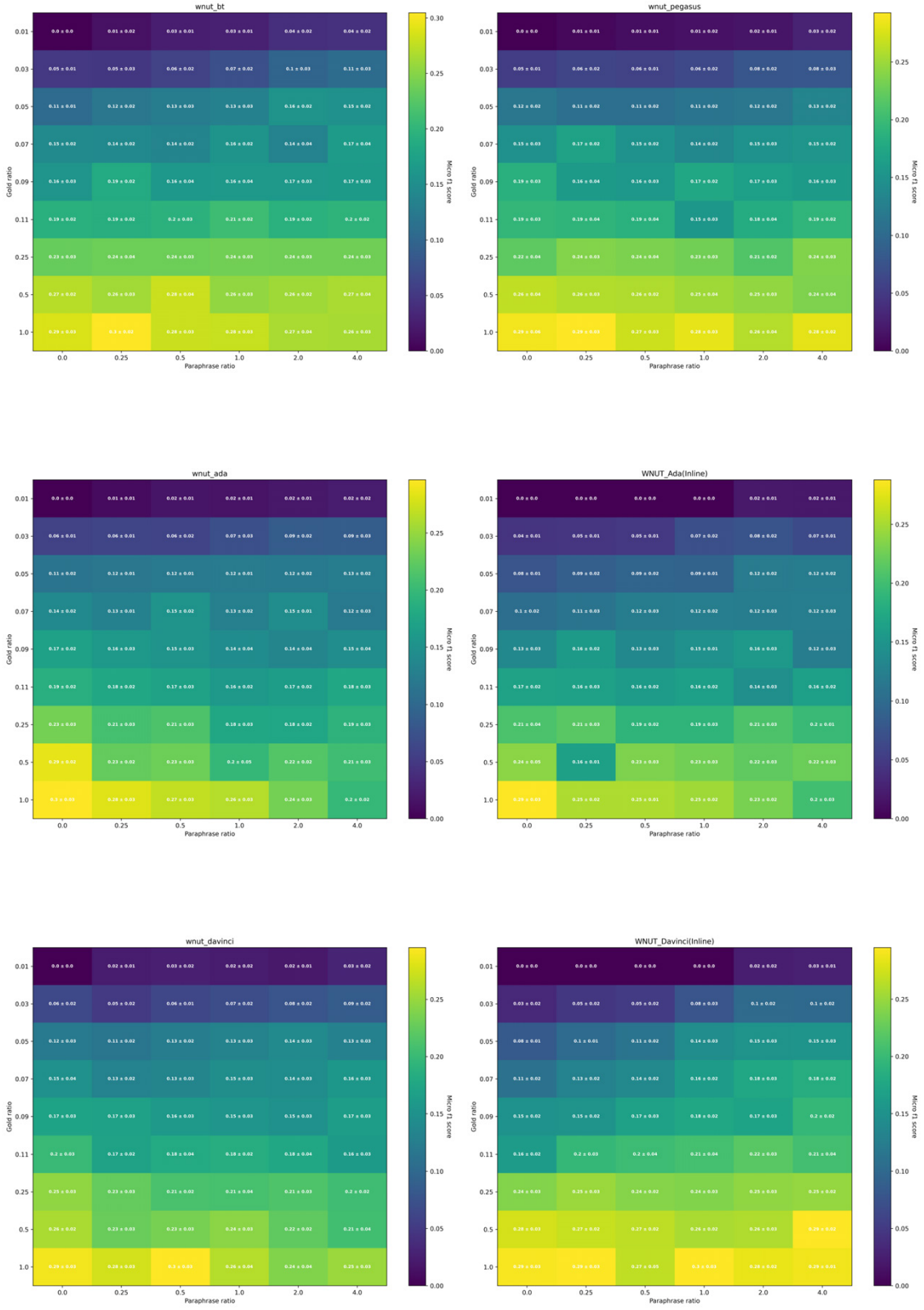


Figure 12: NER performance for all paraphrases on the Unseen Entity Set of Wnut17.

A.11 Ontonotes Unseen Entity Set F1

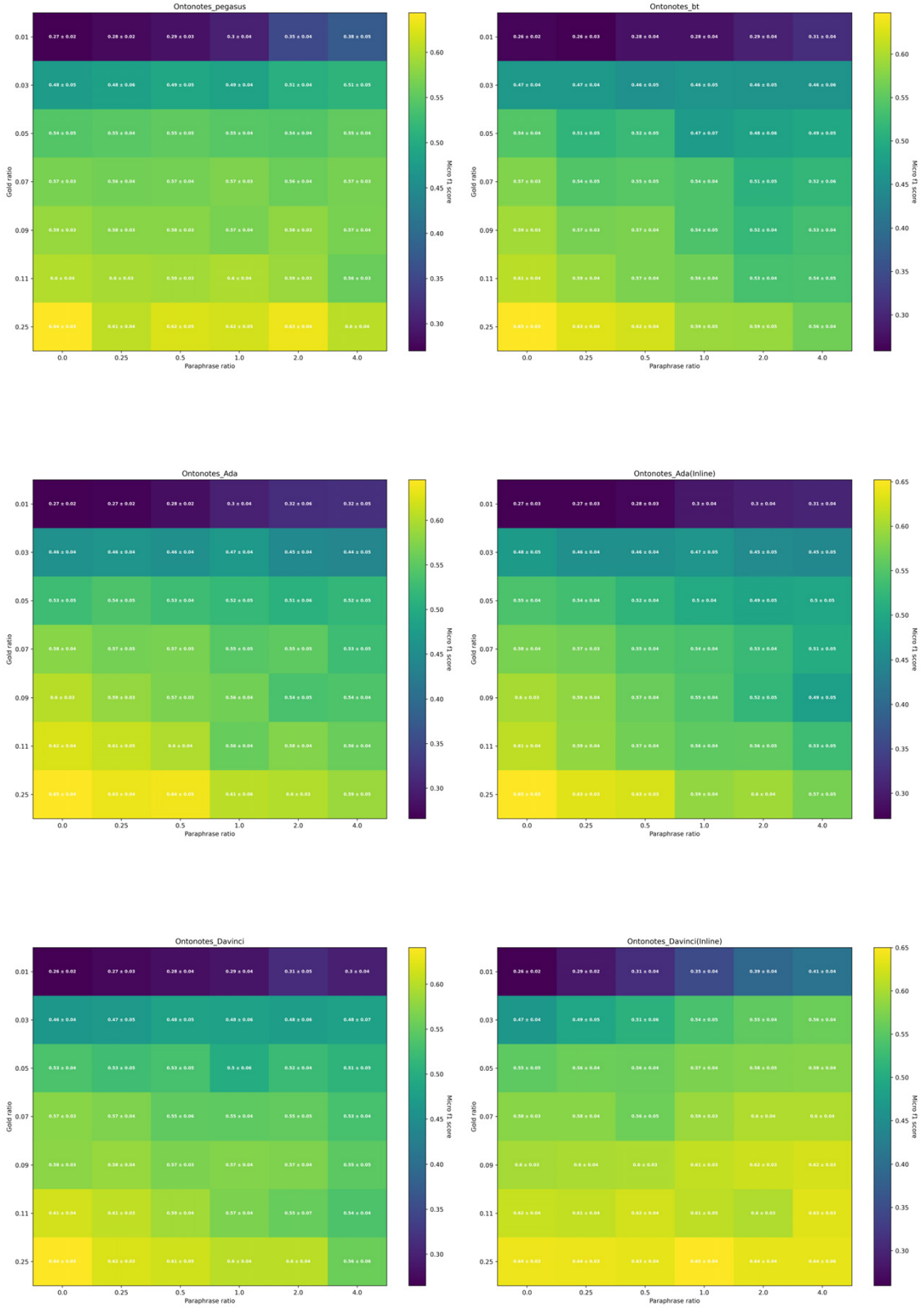


Figure 13: NER performance for all paraphrases on the Unseen Entity Set of Ontonotes.

A.12 MIT Restaurants Unseen Entity Set F1

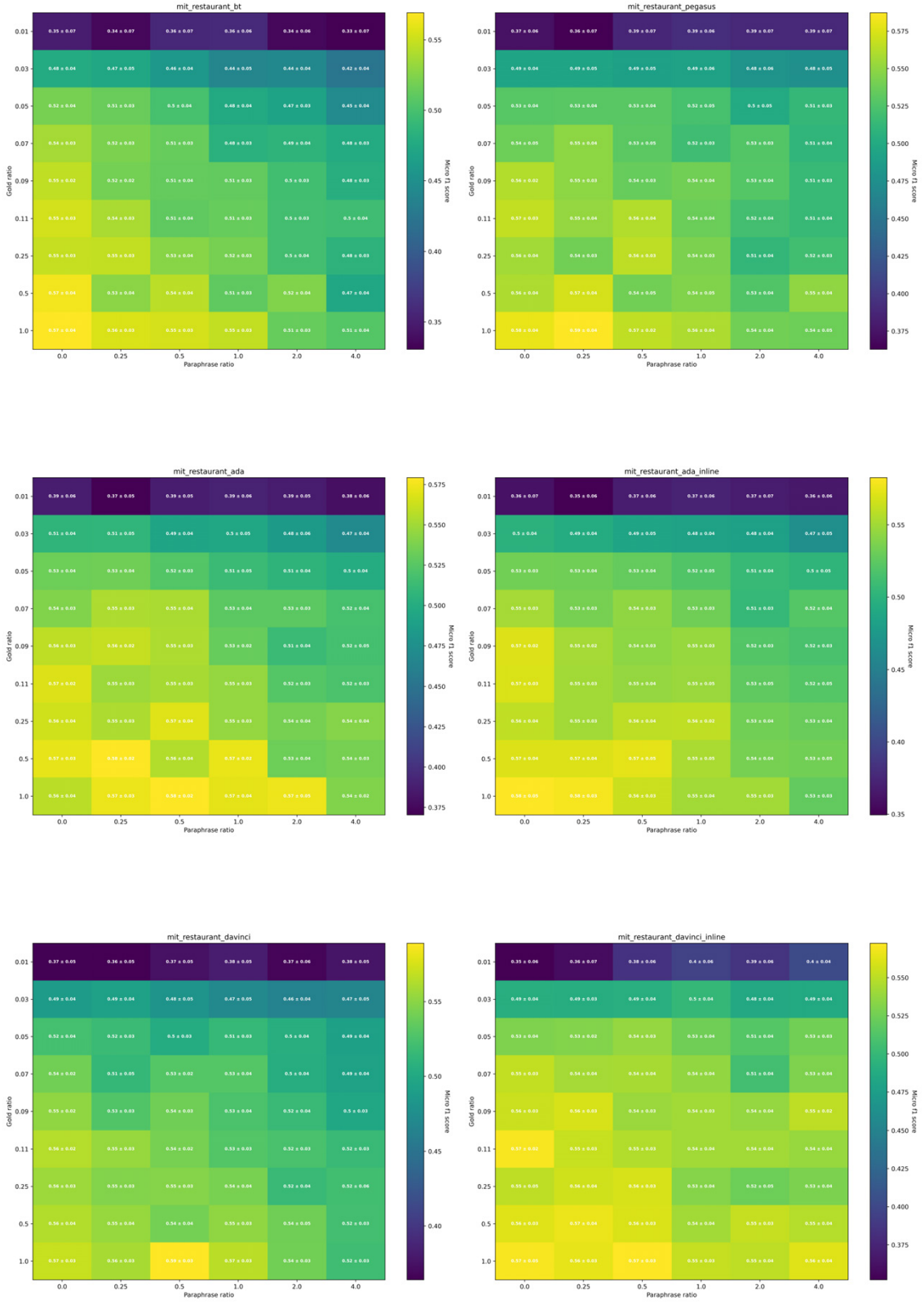


Figure 14: NER performance for all paraphrases on the Unseen Entity Set of MIT Restaurants.

A.13 Bc5cdr Unseen Entity Set F1

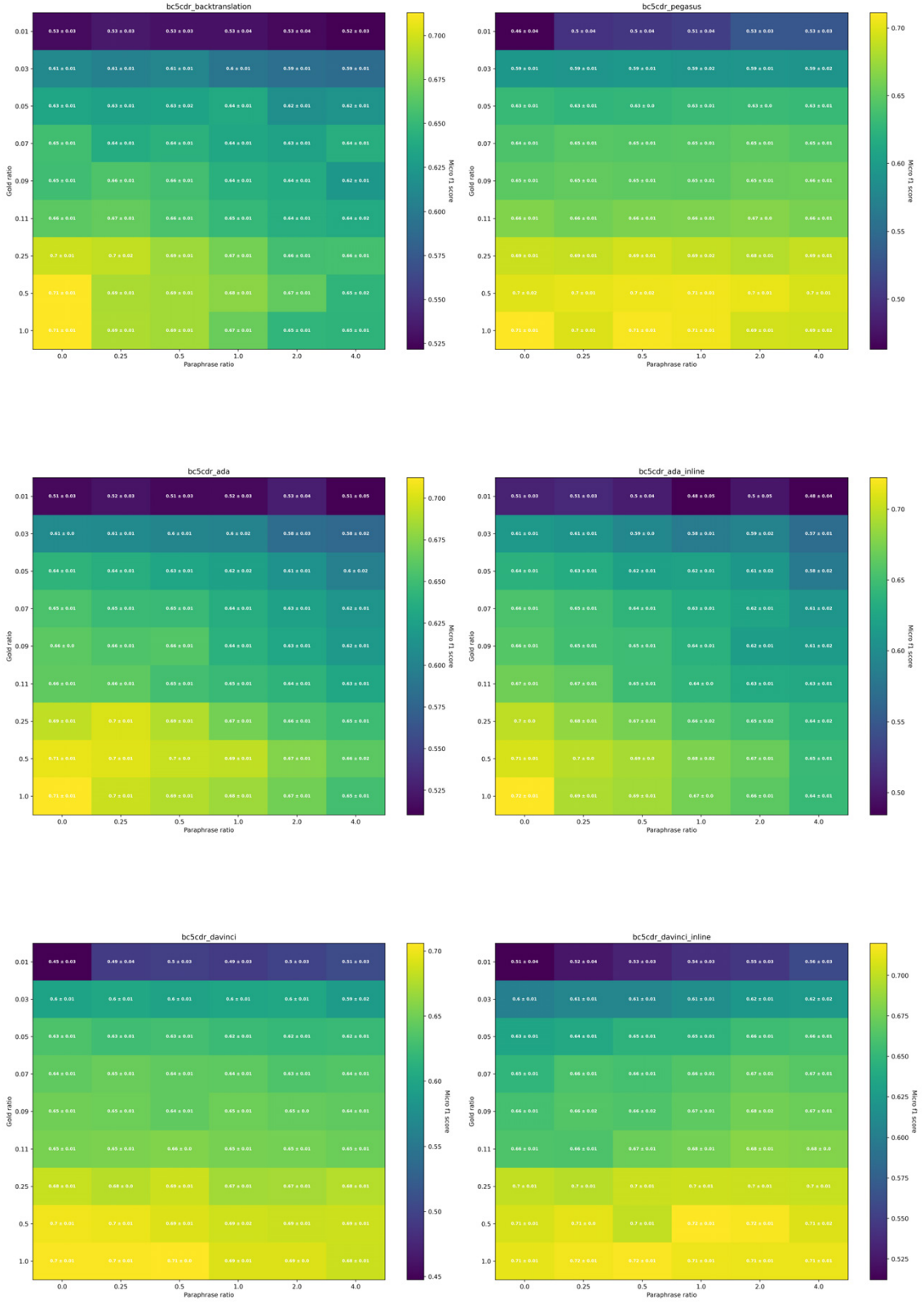


Figure 15: NER performance for all paraphrases on the Unseen Entity Set of Bc5cdr.

A.14 Tweebank Unseen Entity Set F1

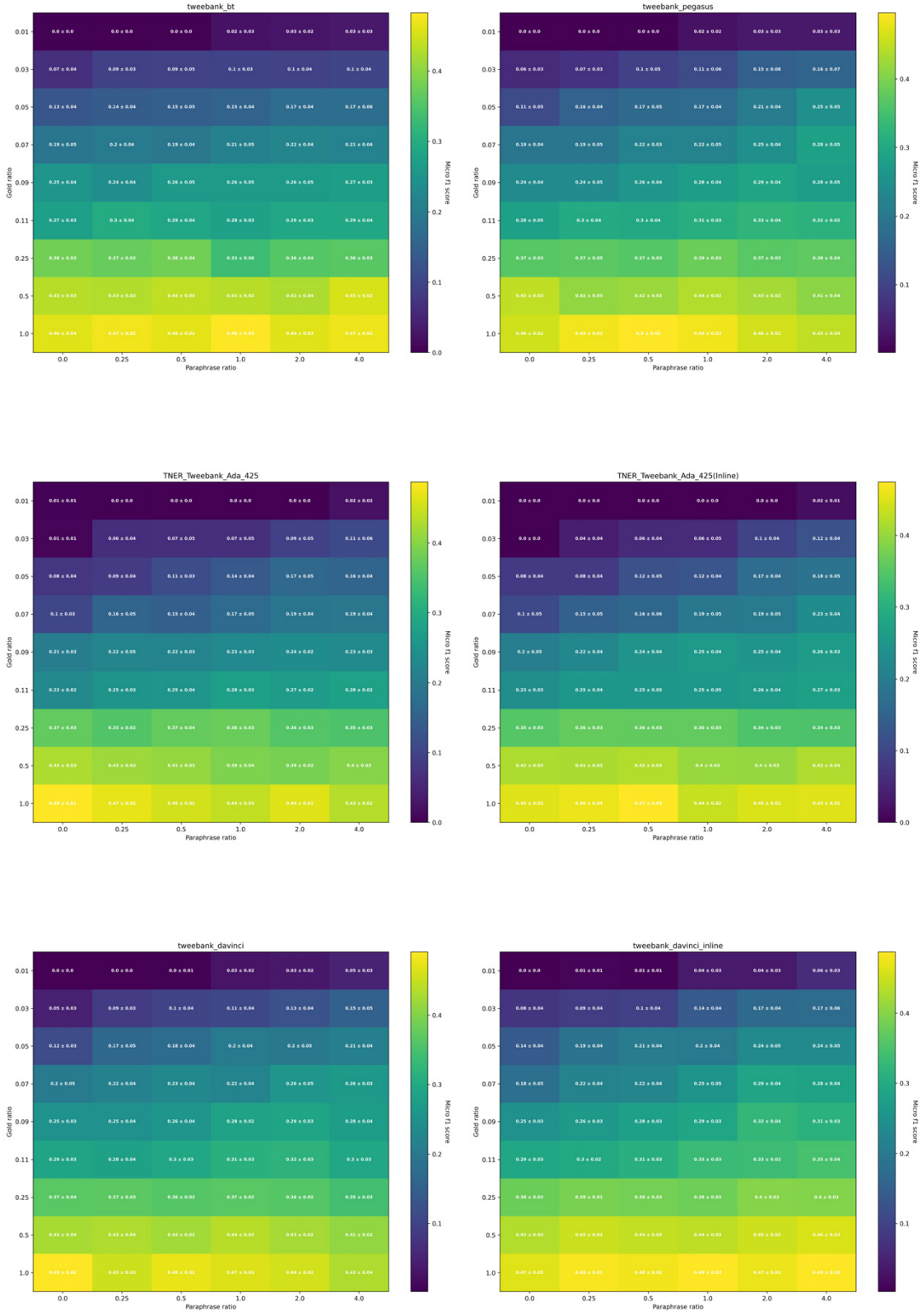


Figure 16: NER performance for all paraphrases on the Unseen Entity Set of Tweebank.

A.15 DaV-B with Mention Replacement Unseen Entity Set F1

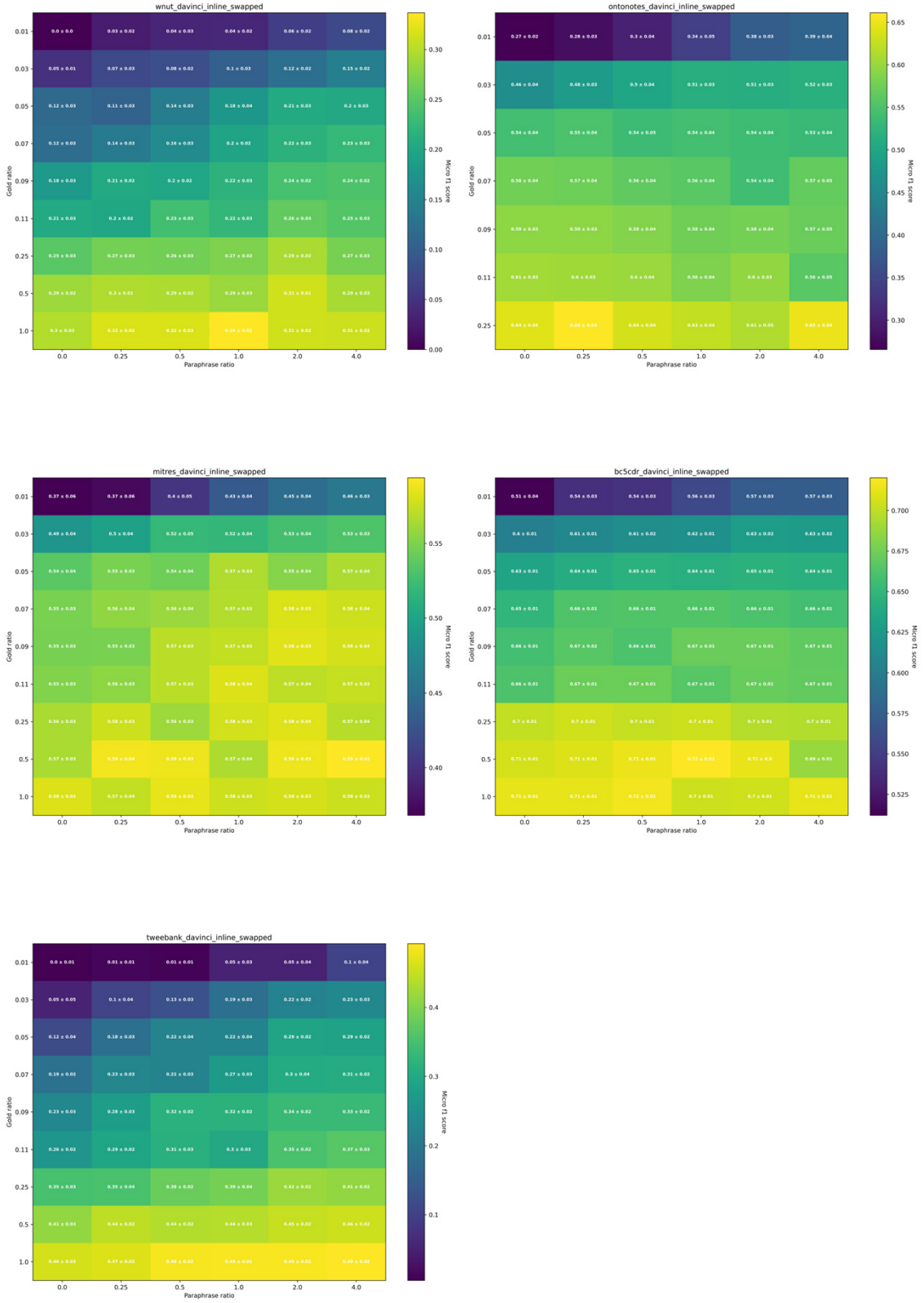


Figure 17: NER performance for DaV-B on the Unseen Entity Set of all Datasets.

A.16 Dataset Statistics

	Train	Dev	Test
BC5CDR	5,228	5330	5,865
Ontonotes	59,924	8,528	8262
MIT-R	6,900	760	1,521
Tweebank	1,639	710	1,201
WNUT-17	2,394	1,009	1,287

Table 10: Dataset statistics.

A.17 Computational budget

Most of our experiments were run on the following GPU machines on AWS: p3.16xlarge, g5.48xlarge, g5.12xlarge and g5.24xlarge. The main fine tunning experiments across G/P ratios took 1-4 days per dataset, depending on the size of the dataset, and the machine used for fine tuning.

Paraphrase generation using GPT-3 DaVinci model took less than a day for most datasets. Ontonotes took roughly a day. Similar time was spent when generating mention replacements.

A.18 Software Acknowledgements

This work would be much harder without the use of several software packages including, but not limited to Pytorch (Paszke et al., 2019), Huggingface transformers (Wolf et al., 2020) and associated software ecosystem (Huggingface datasets), Scipy (Virtanen et al., 2020), Pandas (McKinney et al., 2011), Numpy (Harris et al., 2020), Scikit-learn (Pedregosa et al., 2011), and OpenAI models and Python library.

ACL 2023 Responsible NLP Checklist

A For every submission:

☐ A1. Did you describe the limitations of your work?
Left blank.

☐ A2. Did you discuss any potential risks of your work?
Left blank.

☐ A3. Do the abstract and introduction summarize the paper's main claims?
Left blank.

☐ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☐ Did you use or create scientific artifacts?

Left blank.

☐ B1. Did you cite the creators of artifacts you used?
Left blank.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Left blank.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Left blank.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Left blank.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Left blank.

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Left blank.

C ☐ Did you run computational experiments?

Left blank.

☐ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Left blank.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☐ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Left blank.

☐ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Left blank.

☐ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Left blank.

D ☐ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Left blank.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Left blank.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Left blank.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Left blank.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Left blank.