Abstract

Physical measurements constitute a large portion of numbers in academic papers, engineering reports, and web tables. Current benchmarks fall short of properly evaluating numeracy of pretrained language models on measurements, hindering research on developing new methods and applying them to numerical tasks. To that end, we introduce a novel task, Masked Measurement Prediction (MMP), where a model learns to reconstruct a number together with its associated unit given masked text. MMP is useful for both training new numerically informed models as well as evaluating numeracy of existing systems. To address this task, we introduce a new Generative Masked Measurement (GeMM) model that jointly learns to predict numbers along with their units. We perform fine-grained analyses comparing our model with various ablations and baselines. We use linear probing of traditional pretrained transformer models (RoBERTa) to show that they significantly underperform jointly trained number-unit models, highlighting the difficulty of this new task and the benefits of our proposed pre-training approach. We hope this framework accelerates progress towards building more robust numerical reasoning systems in the future.  

1 Introduction

Many natural language processing tasks require a deep understanding of numbers – for example, reading comprehension (Ran et al., 2019), textual entailment (Sammons et al., 2010; Roy, 2017) and hybrid table tasks such as fact-verification (Chen et al., 2020) or question answering (Chen et al., 2021). Masked number prediction (MNP) is a popular pretraining objective to imbue language models with numerical understanding and evaluate existing models for their numerical capacity. 

As an example of MNP, given the sentence “Cats have [##NUM] paws.” a model learns to predict the number 4. While appropriate for numerical commonsense, MNP is deficient when it is used to predict measurements. Measurements, such as 2 meters or 13.2 square miles, are a special class of particularly common numbers in text that have a well-defined and typed system of units. Given a simple question: “How long did Alex Honnold climb for?”, a single number alone is an insufficient answer since it is meaningless without the unit. Answers like 1000 meters or 4 hours could both suffice.

Current MNP systems do not jointly reason about numbers with units. It is reasonable to expect that pretrained models like BERT could leverage information of units directly as text without

---

1 We will release our trained models and data-splits upon acceptance on Github.
any special treatment. However, in preliminary experiments we find that this yields poor numerical abilities (see Appendix B). Furthermore, including units as text directly raise more questions: should we evaluate using all units (meters, feet, inches)? Should we equally weight across the units? Current models have no opinion about which unit is appropriate because they are not required to make unit predictions during training. Together, this indicates that current training objectives do not capture sufficient representations of measurements and that a direct application of MNP to evaluate numeracy of measurements is ill-suited.

To address these shortcomings, we propose the more challenging task of Masked Measurement Prediction (MMP) along with a new model. In this task, a model must reconstruct both the number together with the correct unit. In Figure 1 we show how in a MMP setting our model generates a dimension (“Length”), a number in metric log-space (“3.00”), the unit (“feet”) and then uses the conversion factor (“3.28”) to deterministically output the full measurement (“3280 feet”). This example illustrates a key distinction in that our model is flexible and can generate non-metric measurements (feet) but evaluates numerical prediction in canonical units (meters).

MMP is useful for two reasons: 1) as a way to train models to give them better numeracy 2) as a new kind of evaluation that allows for a much more fine-grained analysis of reasoning over numerical quantities. The task of measurement estimation decouples the different aspects of numeracy allowing for a more interpretable and thorough analysis of numerical reasoning. We introduce a new evaluation benchmark for MMP based on WikiConvert (WiCo) (Thawani et al., 2021a), a large scale dataset of English Wikipedia sentences with ground truth measurement annotations. We compare the performance of our models on their ability to accurately predict the dimension, unit, and value of a measurement. We employ a large pretrained transformer model as our textual encoder and examine the performance of different discriminative, generative, and latent variable models along with several ablations. Our contributions are as follows:

- We show that linear probing of existing pretrained models on MMP significantly under-performs fully finetuned models.
- We train a model that reasons jointly about numbers and units which predicts numbers 8.1 times more accurately than the probed pretrained models.
- We find our best performing generative model outperforms human annotators on two evaluations, achieving 7.4-7.8% better dimension accuracy and 33.5-39.9% better unit accuracy. Furthermore, this model predicts a number closer to ground truth than our annotators 66.2-78.8% of the time.

Furthermore there are numerous applications of better measurement prediction and unit reconstruction such as in table to text generation (Moosavi et al., 2021), answering numerical queries (Sarawagi and Chakrabarti, 2014; Ho et al., 2019) or for improving e-commerce product search(Arici et al., 2021). We hope that Masked Measurement Prediction becomes a standard benchmarking tool from which we can gain insight how to best incorporate new numeracy modeling techniques as well as evaluate existing models.

2 Models

2.1 Background + Notation

The International System of Units (SI) defines seven fundamental dimensions (Length, Time, Mass, etc.) and seven corresponding base SI units (meters, seconds, kilograms, etc.). The SI system is the most widely adopted measurement standard
and is used internationally in domains such as commerce, finance, logistics, and science. We designate \( \mathcal{D} \) to be the set of composite dimensions obtained from (and including) the fundamental dimensions. Values of \( \mathcal{D} \) include velocity and power. We let \( \mathcal{U} \) be the set of all units: the various ways to describe dimensions. For example, units of Length include meters and miles. Each training example consists of a real number \( y \), a dimension \( d \in \mathcal{D} \), a unit \( u \in \mathcal{U} \), and the remainder of the sentence \( S \). In MMP, our task is to predict \( y \), \( d \), and \( u \) given only \( S \). In the next sections we describe our generative model designed for MMP followed by the ablations we consider.

2.2 Model

Measurements have complex semantic meanings, shaped by many standards, particular instruments, and natural world phenomena. Consider a text concerning rainfall. From a dimensional analysis perspective, the units \( \text{inches per year (in/y)} \) and \( \text{meters per second (m/s)} \) share the same dimension \( \text{velocity} \). However, mentioning \( \text{in/y} \) usually implies that the text is discussing total rainfall in a region. Likewise, the use of \( \text{m/s} \) suggests that the text is examining the speed of falling rain droplets. To capture this complexity, we consider a generative model that learns the joint distribution of the number, dimension, and unit.

We now describe the generative process of our full model. To start, conditioned on \( S \), our model samples a discrete dimension variable \( \mathcal{D} \). Then conditioned on the sampled dimension, our model samples a discrete unit variable \( \mathcal{U} \) compatible with the dimension. For example, conditioned on the dimension \( \text{velocity} \) our model will output a distribution over the units of velocity such as \([\text{miles per hour}, \text{meters per second}, \text{inches per year}]\) as opposed to all of \( \mathcal{U} \). We then separately predict a distribution on the canonicalized measurement, \( \tilde{Y} \), which is the numerical quantity represented in a base canonical (metric) unit like meters. During inference time, we use the highest scoring dimension and unit and choose the proper conversion factor to deterministically produce the final number \( y \) represented in the predicted unit. We refer to this Generative Masked Measurement model as \( \text{GeMM} \), where the joint \( p(\mathcal{D}, Y, U|S) \) is given by the following equation:

\[
p(\mathcal{D}|S) \times p(U|\mathcal{D}, S) \times p(Y|S)
\]

We show the graphical model of \( \text{GeMM} \) in Figure 2. We also consider, \( \text{GeMM} \) a slight variant where we have a direct dependence between the unit and number prediction with a joint equal to:

\[
p(\mathcal{D}|S) \times p(U|\mathcal{D}, S) \times p(Y|U, S)
\]

2.3 Discrete Latent Dimension Model

We also consider an unsupervised generative model which treats the dimension as a discrete latent variable. We use the same number of dimension classes \(|\mathcal{D}|\) and train to maximize the log-likelihood of the observed \( Y \). We refer to this model as \( \text{Lat-Dim} \) and is characterized by:

\[
p(Y|S) = \sum_{\mathcal{D}} p(\mathcal{D}|S) \times p(Y|\mathcal{D}, S)
\]

To evaluate this model we build a contingency matrix of the predicted classes and using a linear solver find the best mapping between our predicted and true dimensions. We can then apply this mapping to the model predictions and calculate classification metrics for dimension prediction.

2.4 Model Ablations

We also consider several model ablations of \( \text{GeMM} \). Our first ablation is \( \text{GeMM} \) which models \( p(\mathcal{D}|S) \). The second, \( \text{GeMM} \), learns the distribution \( p(U, D|S) = p(\mathcal{D}|S) \times p(U|D, S) \). The third, \( \text{GeMM} \), models \( p(Y, D|S) = p(\mathcal{D}|S) \times p(Y|D, S) \). Our final ablation is \( \text{GeMM} \) which learns \( P(Y|D) \) directly.

2.5 Model Architectures

For our textual encoder, we use the Huggingface Transformers (Wolf et al., 2020; Liu et al., 2019) implementation of RoBERTa, a pretrained 12-layer transformer. We refer to this text encoder as \( T \) such that given a sentence \( S \), our model outputs a 768-dimensional vector \( h_T \). We use a single linear layer, \( W_S \in \mathbb{R}^{768 \times M} \), to project \( h_T \) to \( h \) and treat the dimension \( M \) as a hyper-parameter. To form a distribution over the real number line \( \mathbb{R} \) we use a Log-Laplace model, a competitive model used in the numeracy literature (Spokoyny and Berg-Kirkpatrick, 2020; Thawani et al., 2021a; Zhang et al., 2020). This is equivalent to \( L_1 \) regression in log-space and yields the following loss function where \( Y^\ast \) and \( Y \) are predicted and ground truth numbers, respectively:

\[
\log P(Y|S) = |\log Y^\ast - \log Y| + \log \left| \frac{1}{Y} \right|
\]
<table>
<thead>
<tr>
<th>Split</th>
<th>Examples</th>
<th>Max #</th>
<th>Min #</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>919,237</td>
<td>5.5E+36</td>
<td>1E-06</td>
</tr>
<tr>
<td>Train</td>
<td>728,629</td>
<td>5.5E+36</td>
<td>1E-06</td>
</tr>
<tr>
<td>Val</td>
<td>91,110</td>
<td>4.4E+14</td>
<td>1.2E-06</td>
</tr>
<tr>
<td>Test</td>
<td>91,092</td>
<td>1.6E+21</td>
<td>1.8E-06</td>
</tr>
</tbody>
</table>

Table 1: Summary statistics for Wiki-Convert. The median number of characters and tokens per example is 106 and 33, respectively.

As shown in Figure 1, we project $h$ with a linear layer $W_D \in \mathbb{R}^{M \times |D|}$ to obtain a distribution over $D$. We then use a separate linear layer, $W_U \in \mathbb{R}^{M \times |U|}$, to project $h$ and obtain a distribution over $U$. To predict $Y$, we project $h$ with a linear layer $W_Y$. In the case of GeMM, we let $W_Y \in \mathbb{R}^{M \times |D|}$ in order to parameterize a mean of a Log-Laplace distribution for each dimension in $D$. For GeMM, we set $W_Y \in \mathbb{R}^{M \times |U|}$ to output the mean of a Log-Laplace distribution for each unit in $U$ and the remaining models, we set $W_Y \in \mathbb{R}^{M \times 1}$ resulting in a single mean of a Log-Laplace distribution. For training, we use cross-entropy loss for the dimension and unit distributions, and the loss from the equation above for number prediction.

### 3 Dataset

We train and evaluate our models on WiCo (Thawani et al., 2021a), a dataset of English Wikipedia sentences where the number and unit in each sentence are human-annotated. We canonicalize the units and map each to a single dimension. For example both feet per second and miles per hour map to velocity. We show the distribution of all measurements and lengths in Figure 3. The resulting dataset consists of 919,237 sentences with annotated (number, unit, dimension) triples. We provide more details on the data in Appendix A.

### 4 Experiments

We train all models using a batch size of 200 and use the AdamW (Loshchilov and Hutter, 2019) optimizer with a learning rate of $1e^{-3}$ and a linear warm-up schedule of 500 steps. We use the “cold” symbol to indicate that we freeze the transformer parameters for training. For all frozen models we use a log frequency weighted cross-entropy due to the highly imbalanced classes as well as a higher learning rate of $1e^{-3}$. We employ early stopping with a patience of five epochs on validation score.

To evaluate the performance of our models, we report the macro averaged F1 score for dimension and unit prediction and log-mae to evaluate number prediction. We define log-mae in Equation 2 where $Y$ is the predicted number and $Y^*$ is the ground truth number. As a simple baseline for dimension and unit prediction, we employ majority class voting. For number prediction we use the median of all the numbers in the training set.

$$log-mae = \frac{1}{|D_{test}|} \sum_{d \in D_{test}} |\log_{10} Y^* - \log_{10} Y|$$

(2)

#### 4.1 Few-Shot

To study the degree to which current pretrained models capture different aspects of numeracy, we consider the following few-shot experiment. We sample a balanced dataset of dimensions where each class gets 10, 40, 70, or 100 labeled examples. We train GeMM and GeMM on the few-shot task where the pretrained text encoder $T$ parameters are frozen and compare their performance against full fine-tuning. Due to the high variance of GeMM, we report the average of three random seeds. In Table 2 and Table 3 we show results of GeMM and GeMM respectively.

Table 2: Results (measured by F1 ↑) of our few-shot experiment on dimension classification (probing $p(D|S)$). $x$-shot implies the model is trained on $x$ labeled examples per dimension. GeMM indicates an ablation of GeMM where $Y$ and $U$ are not modeled. cold indicates the model’s parameters are frozen during training.

<table>
<thead>
<tr>
<th>Model</th>
<th>10-shot</th>
<th>40-shot</th>
<th>70-shot</th>
<th>100-shot</th>
</tr>
</thead>
<tbody>
<tr>
<td>GeMM</td>
<td>15.5</td>
<td>50.0</td>
<td>52.5</td>
<td>53.4</td>
</tr>
<tr>
<td>GeMM</td>
<td>42.5</td>
<td>51.2</td>
<td>57.6</td>
<td>60.5</td>
</tr>
<tr>
<td>Majority</td>
<td>14.3</td>
<td>14.3</td>
<td>14.3</td>
<td>14.3</td>
</tr>
</tbody>
</table>

Table 3: Results (log-mae ↓) of our few-shot experiment on number prediction (probing $p(Y|S)$).

<table>
<thead>
<tr>
<th>Model</th>
<th>10-shot</th>
<th>40-shot</th>
<th>70-shot</th>
<th>100-shot</th>
</tr>
</thead>
<tbody>
<tr>
<td>GeMM</td>
<td>1.94</td>
<td>1.82</td>
<td>1.72</td>
<td>1.75</td>
</tr>
<tr>
<td>GeMM</td>
<td>1.70</td>
<td>1.56</td>
<td>1.43</td>
<td>1.41</td>
</tr>
<tr>
<td>Median</td>
<td>1.99</td>
<td>1.99</td>
<td>1.99</td>
<td>1.99</td>
</tr>
</tbody>
</table>

Although performance improves with more data, the frozen models significantly underperform their
Figure 3: Histograms of WiCo numbers binned by base-10 exponent. All numbers are canonicalized to their SI form. **Left:** All numbers labeled by dimension. **Right:** Numbers in the length dimension labeled by unit.

<table>
<thead>
<tr>
<th>Model</th>
<th>Probing Type</th>
<th>Val</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Majority</td>
<td>-</td>
<td>33.1</td>
<td>33.1</td>
</tr>
<tr>
<td>GeMM</td>
<td>p(D</td>
<td>S)</td>
<td>69.1</td>
</tr>
<tr>
<td>GeMM</td>
<td>p(D</td>
<td>S)</td>
<td>88.0</td>
</tr>
<tr>
<td>GeMM</td>
<td>p(D</td>
<td>S)</td>
<td>87.0</td>
</tr>
<tr>
<td>Lat-Dim</td>
<td>p(D</td>
<td>S)</td>
<td>87.2</td>
</tr>
<tr>
<td>GeMM</td>
<td>p(D</td>
<td>S)</td>
<td>84.7</td>
</tr>
</tbody>
</table>

Table 4: Results (F1↑) for dimension prediction conditioned on S only. GeMM indicates a variant of GeMM where Y is dependent on U (in addition to S).

<table>
<thead>
<tr>
<th>Model</th>
<th>Probing Type</th>
<th>Val</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Majority</td>
<td>-</td>
<td>8.9</td>
<td>9.0</td>
</tr>
<tr>
<td>GeMM</td>
<td>p(U</td>
<td>D, S)</td>
<td>29.8</td>
</tr>
<tr>
<td>GeMM</td>
<td>p(U</td>
<td>D, S)</td>
<td>52.9</td>
</tr>
<tr>
<td>GeMM</td>
<td>p(U</td>
<td>D, S)</td>
<td>51.5</td>
</tr>
<tr>
<td>GeMM</td>
<td>p(U</td>
<td>D, S)</td>
<td>49.3</td>
</tr>
</tbody>
</table>

Table 6: Results (F1↑) on unit prediction conditioned on the true dimension and text. Ablations are above the double horizontal line.

4.2 Dimension Prediction

We train our models and their ablations on the full dataset and measure their performance on dimension prediction. In Table 4, we show the results of dimension prediction conditioned on S. We observe that the performance gap between the frozen and unfrozen GeMM grows to 19.5 F1 on the test split despite training on 3 orders of magnitude more training data than the few-shot setting.

By using Bayes’ rule, we perform dimension prediction conditioned on both S and Y and show our results in Table 5. We observe that both models show improved dimension prediction ability when supplied with the number with GeMM reaching 96.6 F1 score, an effective error rate reduction of 75%.

4.3 Unit Prediction

We show the unit prediction performance of our models in Table 6. The strongest performing model for unit prediction was GeMM with a F1 score of 54.9. Again, the frozen GeMM produced a 25.1 lower F1 score than its unfrozen counterpart.

We note that even though the F1 scores on unit prediction are much lower than dimension prediction, they are still significantly better than the majority baseline. Although one can freely substitute a unit with one in the same dimensional class, we tend to be more systematic and choose units that allow for more straightforward human readability or reflect the actual instruments used for measure-
Table 7: Results (log-mae ↓) for number prediction conditioned on $S$. In the second row of GeMM, we select the highest scoring $d^* \in D$ and predict $y$ conditioned on $d^*$ and $S$. In the second row of GeMM, we select the highest scoring $u^* \in U$ and $d^* \in D$ and predict $y$ conditioned on $u^*, d^*$, and $S$. For Lat-Dim, we sum over the latent variable $D$ to predict $y$ conditioned on $S$.

<table>
<thead>
<tr>
<th>Model</th>
<th>Probing Type</th>
<th>Val</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Median</td>
<td>-</td>
<td>1.98</td>
<td>1.97</td>
</tr>
<tr>
<td>GeMM U</td>
<td>$p(\hat{Y}</td>
<td>S)$</td>
<td>1.377</td>
</tr>
<tr>
<td>GeMM U</td>
<td>$p(\hat{Y}</td>
<td>S)$</td>
<td>0.529</td>
</tr>
<tr>
<td>GeMM U</td>
<td>$p(\hat{Y}</td>
<td>D, S)$</td>
<td>0.468</td>
</tr>
<tr>
<td>GeMM U</td>
<td>$p(\hat{Y}, D</td>
<td>S)$</td>
<td>0.517</td>
</tr>
<tr>
<td>Lat-Dim</td>
<td>$p(\hat{Y}, D</td>
<td>S)$</td>
<td>0.545</td>
</tr>
</tbody>
</table>

4.4 Number Prediction

We show the number prediction performance of our models in Table 7. Consistent with our previous experiments, all models outperform GeMM. Furthermore, we observe that not modeling $U$ and $D$ (as is the case in GeMM) increases log-mae, i.e., results in worse numerical prediction. While competitive with GeMM and its variants on number prediction, Lat-Dim cannot predict dimensions with the same efficacy (Table 4).

We also experiment with the setting where GeMM conditionally generates the number for a particular dimension. In this setting, GeMM improves log-mae to 0.469. Extending this setting further, we condition GeMM on both a unit and a dimension to produce the best log-mae among our models: 0.401.

We now revisit our original motivating example: “Alex Honnold climbed for [NUM] [UNIT]”. Assume we want to know the distance of a climb. To do this, we condition GeMM on $D = \text{length}$ and $U = \text{feet}$. If, on the other hand, we want to know the duration of a climb, we change the conditioning to $D = \text{time}$ and $U = \text{hours}$. Now, if we want to know the length of Alex Honnold’s climbing career, we condition GeMM on $D = \text{time}$ and $U = \text{years}$. These examples illustrate the flexibility of GeMM and the importance of jointly modeling numbers, units, and dimensions.

4.5 Quantitative Analysis

4.5.1 Dimensions and Unit

In Figure 4a we visualize a confusion matrix of dimension predictions by GeMM. The low accuracy for electric charge and temperature is attributed to a mislabeling in the dataset. For mass, we find many ambiguous situations where either mass or length are appropriate. See the first row of Table 10 for such an example.

Thus far, we have treated dimensions as distinct classes with no relationships. However, dimensions are compositions of the seven fundamental dimensions. Therefore, dimensions that share fundamental dimensions are more similar than those that do not. To quantify this similarity, we can treat dimensions as a vector where each element represents the exponent of a fundamental dimension. Then to measure the similarity of two dimensions, we take their Manhattan distance. To illustrate, assume there exist only two fundamental dimensions: Length and Time. Let $speed = (1, -1)$ and $length = (1, 0)$ where the first element represents Length and the second represents Time. The Manhattan distance between $speed$ and $length$ is equal to one. In Figure 5, we visualize the Manhattan distance between the predictions of GeMM and ground truth. We observe that there is generally an inverse relationship between error count and the distance of the errors. This observation suggests that our model has learned that some dimensions are more similar than others. This suggestion is reinforced by Figure 4a where misclassifications tend to have small distances from the true dimension. For example, velocity is most often misclassified as length. For unit prediction, we find that most mistakes occur substituting units with ones that have similar magnitudes like feet for meters or kilometers for miles.

4.5.2 Numeracy

In Table 8, we show log-mae by dimension as predicted by GeMM. We note that errors are not uniform across dimensions, predicting areas is 2.2

---

3Sentences with mislabeled Celsius as Coulombs, which may due to wrong annotation between °C and C. Also observed by Elazar et al. (2019)
Figure 4: Confusion matrices for predictions by GeMM U)Y over the validation split. Left 4a: Dimension prediction. Most misclassified dimensions are similar to their ground truth counterparts in terms of Manhattan distance. Right 4b: Unit prediction for examples that share the length dimension. Most misclassified units of length share similar magnitudes to their ground truth units.

Figure 5: Manhattan distance between true and predicted dimensions by GeMM U)Y. We treat dimensions as vectors whose elements are the exponents of the fundamental dimensions that compose a given dimension. Note that the y-axis is in log-scale.

<table>
<thead>
<tr>
<th>Length</th>
<th>Area</th>
<th>Velocity</th>
<th>Mass</th>
<th>Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.37</td>
<td>0.54</td>
<td>0.19</td>
<td>0.55</td>
<td>0.27</td>
</tr>
</tbody>
</table>

Table 8: log-mae ↓ by dimension. It is harder to predict numbers of Area and Mass than other dimensions.

<table>
<thead>
<tr>
<th>Model</th>
<th>Human</th>
<th>Model &gt; Human</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>D</td>
<td>U</td>
</tr>
<tr>
<td>Tech Ann.</td>
<td>96.7</td>
<td>86.2</td>
</tr>
<tr>
<td>AMT Ann.</td>
<td>96.7</td>
<td>77.0</td>
</tr>
</tbody>
</table>

Table 9: Dimension and unit prediction accuracy of our human evaluation experiment. GeMM U)Y outperformed the human annotators in both evaluations. Tech Ann. is over a balanced set of 90 sentences labeled by Technical Annotators. AMT Ann. is over a balanced set of 2,122 sentences annotated by AMT Annotators. The final column shows the model predicted a number closer to ground truth in 66.2-78.8% of the cases.

4.5.3 Human Evaluation

We perform two evaluations of GeMM U)Y against human annotators. In the first evaluation, we compare against the combined effort of three Technical Annotators on a balanced set of 90 sentences randomly sampled from the test set. The annotators worked together to predict the missing dimensions, times harder velocities. We also observe that the magnitudes of errors seem to be positively correlated with the variances observed in Figure 3.
<table>
<thead>
<tr>
<th>#</th>
<th>Text</th>
<th>Dim</th>
<th>Unit</th>
<th>Num</th>
<th>Dim</th>
<th>Unit</th>
<th>Num</th>
<th>Dim</th>
<th>Unit</th>
<th>Num</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Hope is gaff rigged, ‘V’-bottomed and has an [#NUM] [UNIT] centerboard.</td>
<td>Mass</td>
<td>pounds</td>
<td>385.6</td>
<td>Length</td>
<td>feet</td>
<td>2.97</td>
<td>Length</td>
<td>meter</td>
<td>50</td>
</tr>
<tr>
<td>2</td>
<td>Some have been running for over 50 years, each covering about [#NUM] [UNIT].</td>
<td>Velocity</td>
<td>miles/year</td>
<td>0.10</td>
<td>Area</td>
<td>sqkm</td>
<td>2.09E+10</td>
<td>Area</td>
<td>sqmi</td>
<td>2.59E+07</td>
</tr>
<tr>
<td>3</td>
<td>Another medium-sized corvid, the [#NUM] [UNIT] Eurasian magpie (Pica pica) is also amongst the most widely reported secondary prey species for goshawks there.</td>
<td>Mass</td>
<td>grams</td>
<td>0.22</td>
<td>Mass</td>
<td>grams</td>
<td>0.05</td>
<td>Mass</td>
<td>grams</td>
<td>0.2</td>
</tr>
<tr>
<td>4</td>
<td>The twin cylinder, liquid-cooled, in-line two-stroke, [#NUM] [UNIT] Rotax 582 has also been used.</td>
<td>Power</td>
<td>horse-power</td>
<td>47725</td>
<td>Power</td>
<td>horse-power</td>
<td>39248</td>
<td>Power</td>
<td>horse-power</td>
<td>45000</td>
</tr>
<tr>
<td>5</td>
<td>Chrysothamnus may grow up to a [#NUM] [UNIT] tall shrub or subshrub, usually with woody stem bases.</td>
<td>Length</td>
<td>cms</td>
<td>1.2</td>
<td>Length</td>
<td>meters</td>
<td>1.147</td>
<td>Length</td>
<td>meters</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>Kurt Busch was the fastest in the first practice session with a time of 21.372 seconds and a speed of [#NUM] [UNIT].</td>
<td>Velocity</td>
<td>miles/hour</td>
<td>75.1</td>
<td>Velocity</td>
<td>miles/hour</td>
<td>63.584</td>
<td>Velocity</td>
<td>meters/second</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 10: Instances of the MMP task performed during our human evaluation experiment, all numbers are in SI units. In ex. 1, both the model and humans predict the incorrect dimension length instead of mass. The preceding sentence of ex. 2 references “trains” leading both to incorrectly predict area instead of velocity. In ex. 6 the model predicts the speed of the NASCAR driver Kurt Busch’s car whereas the humans had mistaken him for a runner.

In the second evaluation, we compare against Amazon Mechanical Turk (AMT) Annotators on a balanced set of 2,122 sentences randomly sampled from the test set. We show the results for both evaluations in Table 9.

In both evaluations, the model outperforms the human annotators on every task. For dimension prediction, the model led by 7.4-7.8 percentage points. Of the sentences where the dimension was correctly annotated, the model led by 33.5-39.9 percentage points on unit prediction. For sentences where both the model and human correctly predicted the dimension, the model predicted a number closer to ground truth 66.2-78.8% of the time.

### 4.6 Qualitative Analysis

#### 4.6.1 Semantic Head Embeddings

In Figure 6 we plot the t-SNE embeddings of the sentences’ $h$, the output of our text encoder. We label each $h$ with the masked measurement’s true dimension, unit and exponent of the number. In 6a we observe that most embeddings labeled by their true dimension tend to form tight clusters. In 6b we filter to only show embeddings that share the Length dimension and label them by their units. We find that clusters are organized by the relative magnitudes of their units: large (Kilometers, miles), medium (feet, meters), and small (millimeters, inches, centimeters). Further we see that yards appear close to other imperial units of feet and miles. Finally, in 6c when embeddings are binned by the exponent of their values we observe that the left to right direction appears to capture the increasing magnitude of a number.

### 5 Related Work

#### 5.1 Numeracy

Multiple works have probed word embeddings like word2vec, GloVe, FastText (Naik et al., 2019) and contextual embeddings from models like BERT (Wallace et al., 2019; Zhang et al., 2020) or T5 (Pal and Baral, 2021) on a variety of numerical tasks like sorting, numeration, magnitude prediction, and common sense (Lin et al., 2020). Several works have targeted numeracy pretraining using left to right language models (Spithourakis and Riedel, 2018), CNN and RNN based models (Chen et al., 2019), pretrained transformers (Spokoyny and Berg-Kirkpatrick, 2020; Jin et al., 2021), for an overview (Thawani et al., 2021b).

Incorporating synthetic mathematical data augmentations (Geva et al., 2020) has improved question answering while numerical pretraining has been shown to lower masked language modelling perplexity (Thawani et al., 2021a). Either directly or indirectly units have been involved in providing more interpretable explanation of quantities (Changandy and Liang, 2016), solving Fermi problems (Kalyan et al., 2021) and resolving numeric Fused-Heads (Elazar and Goldberg, 2019).
Figure 6: t-SNE visualizations of semantic head embeddings labeled by (left 6a) dimension, (middle 6b) units of length, and (right 6c) number exponent bin. Middle: we observe a clustering of imperial units: feet, yards, miles. Right: we show two directions where magnitudes of length and area measurements increase in value.

5.1.1 Numeracy Benchmarks

Several numeracy benchmarks have been proposed like quantitative reasoning in natural language entailment (Ravichander et al., 2019) and synthetic measurement estimation (Jin et al., 2021). The closest benchmark to our work is the Distribution over Quantities dataset (DoQ) introduced by Elazar et al. (2019). A rule-based method was combined with simple heuristics to build DoQ resulting in its high-coverage albeit also higher noise. Although, WiCo is smaller, it has much higher fidelity since it utilizes a feature used by editors of Wikipedia to automatically convert quantities into different units. Further, WiCo provides the whole sentence as context as opposed to triplets of words. Zhang et al. (2020) use artificial templates to probe models on DoQ and find little difference between numerically pretrained and frozen embeddings such as ELMo. In contrast, our findings show there is a significant gap on WiCo between fully finetuned models and their frozen counterparts.

6 Limitations

The pretrained RoBERTa model we use in experiments was likely pretrained on data that included WiCo. Thus, it is reasonable to be concerned about inflated test performance. That said, the task we consider is distinct from the self-supervised task used to pretrain RoBERTa (i.e. masked word classification vs. masked number regression). Further, our experiments on directly probing RoBERTa to predict masked numbers and units showed poor performance – indicating, perhaps, that even if RoBERTa’s pre-training set did include WiCo, RoBERTa did not memorize aspects of our test set relevant to masked number prediction, partially mitigating these concerns.

The human evaluation studies we conducted are a quite limited ‘guessimating’ task. The human annotators were not allowed to use any external information from searching the internet or looking up answers in knowledge-bases. Their total average completion time per question was 33 seconds. Furthermore, many annotators may not have strong intuition about measurements with unfamiliar and uncommon unit types. For these reasons it is not surprising that our models outperform the human annotators in this limited experiment. However, these human evaluation studies do help calibrate the difficulty of the MMP task on WiCo.

7 Conclusion

In this work we propose Masked Measurement Prediction, a new task that requires models to jointly predict masked numbers and units in running text. We motivate this task as an important extension of existing masked number-only prediction tasks that addresses their limitations and allows for better evaluation of numeracy in NLP models. In our study, we show that probing of traditional pretrained transformers exposes a gap in their understanding of contextualized quantities. Through careful quantitative and qualitative analysis of our new model, which directly reasons about underlying units and dimensions, we find that it is possible to learn good representations of measurements. For future work we aim to extend this dataset to cover more existing standardized units from organizations such as UNECE.4 We hope our MMP task encourages research into further development of better numeracy methodologies.

4United Nations Economic Commission for Europe
References

Tarik Arici, Kushal Kumar, Hayreddin Çeker, K K Saladi, and Ismail B. Tutar. 2021. Solving price per unit problem around the world: Formulating fact extraction as question answering. In KDD TrueFact Workshop.


Yanai Elazar, A. Mahabal, Deepak Ramachandran, Tania Bedrax-Weiss, and Dan Roth. 2019. How large are lions? inducing distributions over quantitative attributes. In ACL.

Mor Geva, Ankit Gupta, and Jonathan Berant. 2020. Injecting numerical reasoning skills into language models. In ACL.

Vinh Thinh Ho, Yusra Ibrahim, Konnika Pal, Klaus Bergerich, and Gerhard Weikum. 2019. Qsearch: Answering quantity queries from text. In SEMWEB.


Mark Sammons, V.G.Vinod Vidyasaran, and Dan Roth. 2010. “ask not what textual entailment can do for you...”. In ACL.


We train and evaluate our models on Wiki-Convert (WiCo) (Thawani et al., 2021a), a dataset of English Wikipedia sentences where the number and unit in each sentence are human-annotated. The built-in template in Wikipedia can ensure the text contains numbers and units. For example, `{{convert|2|km|mi}}` displays as 2 kilometres (1.2 mi). By searching within Wikipedia articles for the use of this template, the authors of WiCo automatically extract human-annotated numbers. To perform unit canonicalization, we use Pint whenever the mapping is unambiguous. In the ambiguous case, we manually inspect the sentence and perform the mapping. For example, we map the unit `sqmi` in WiCo to `square miles` to let pint perform unit canonicalization. Table 10 shows examples of the extended dataset. The original dataset contains 924,473 sentence. The median sentence length is 106 characters, with 29,597 sentences has a length shorter than 20 characters. We provide statistics of the data in Table 1. For preprocessing we exclude sentences which have more than 64 tokens to have efficient computing memory or where the number is negative for simplicity. According to Thawani et al. (2021a) WiCo, “... has been extracted from Wikipedia dumps, which are licensed under the GNU Free Documentation License (GFDL) and the Creative Commons Attribution-Share-Alike 3.0 License.” Thawani et al. (2021a) constructed WiCo with the intent that it be used to further numeracy NLP research. Our use of WiCo is aligned with its authors’ goals.

### B MLM Preliminary Unit Probe

We perform a preliminary unit probe shown in Table 11. The model predicts vastly different numbers when conditioned on different units. We observe a mean of 3086.8 and a standard deviation of 5820 for all the converted metric output.

### C Experiments

We train our model GeMM U (GeMM U) on a single Nvidia GeForce RTX 2080 Ti for 4 hours and 14 minutes with a total parameter of 124,696,538.

#### C.1 Quantitative Analysis

In Figure 7, we show log-mae is relatively small for small magnitude units, which means predicting
numbers for small magnitude units is easier than predicting numbers for their larger counterparts.

In Figure 4, we show confusion matrices of dimension and unit predictions by GeMM.

D Human Annotators

D.1 Evaluation 1

The Technical Annotators have diverse scientific backgrounds ranging from chemistry, earth sciences, and computer science. One annotator is a native Chinese speaker, and two are native English speakers.

D.2 Evaluation 2

In Figure 8 we show the instructions provided along with the interface we designed for our MMP task. While the workers’ geographic location were not provided to us by Mechanical Turk, we aimed to compensate the workers above the US federal minimum wage of $7.25. We paid workers $0.15 per annotation with an average completion time of 33 seconds. This equates to an hourly rate of $12.80 after Mechanical Turk fees. Other demographic information is only provided by Mechanical Turk for an extra fee.

E Ethical Considerations

Like any system that makes predictions, those made by GeMM are not necessarily accurate and may be used by malicious actors to generate fake information to mislead their audience. Additionally, GeMM is an extension of RoBERTa and therefore inherits the biases learned during the training of RoBERTa. Our work focuses exclusively on English and Arabic numerals. As noted by Thawani et al. (2021a), the units in WiCo are heavily biased towards European and American units as they are over-represented in English Wikipedia.
Figure 8: **Left:** Instructions for labeling task. **Right:** we show the interface used by the labelers

**Labeling Instructions**

Instructions: For each sentence please give your best estimate for the number in the units. Do not look things up, certain questions are ambiguous and that’s okay. Really important the number will be interpreted in the units that you select. For number please just input the digits and decimals points without any spaces or commas.

Some examples:

1. My car weighs [NUM][UNIT]: Answer: Dimension=Mass, Unit=ton, Value=1
2. My brother is [NUM][UNIT] tall: Answer: Dimension=Length, Unit=ft, Value=5.8
3. My house is [NUM][UNIT] large: Answer: Dimension=area, Unit=ft², Value=1300.4

**Please Guess the Dimension**

- Length
- Mass
- Area
- Velocity
- Power

**Please Guess the Number**

100

**and the Units**

- Quick Filter
  - meters (m)
  - miles (mi)
  - centimeters (cm)
  - kilometers (km)
  - millimeters (mm)
  - feet (ft)
  - inches (in)
  - yards (yd)