Representing Numbers in NLP: a Survey and a Vision

Avijit Thawani and **Jay Pujara** and **Pedro Szekely** and **Filip Ilievski**
University of Southern California
Information Sciences Institute
{thawani,jpujara,pszekely,ilievski}@isi.edu

Abstract

NLP systems rarely give special consideration to numbers found in text. This starkly contrasts with the consensus in neuroscience that, in the brain, numbers are represented differently from words. We arrange recent NLP work on numeracy into a comprehensive taxonomy of tasks and methods. We break down the subjective notion of numeracy into 7 subtasks, arranged along two dimensions: granularity (exact vs approximate) and units (abstract vs grounded). We analyze the myriad representational choices made by 18 previously published number encoders and decoders. We synthesize best practices for representing numbers in text and articulate a vision for holistic numeracy in NLP, comprised of design trade-offs and a unified evaluation.

1 Introduction

Numbers are an integral part of text. To understand a simple sentence like *I woke up at 11*, we need not just literacy but also numeracy. We must decode the string *11* to the quantity `11` and infer `11` to denote a time of the day, probably `11 a.m.` We need commonsense to reason that 11 a.m. is quite late in the morning. This interpretation of `11` is strongly contextual, as *I earn $11 per month* evokes different units and value expectations. Note how the semantics remains the same for both sentences if `11` was replaced by `10`, i.e., the context is tolerant to some variability.

Numbers are everywhere. Reasoning with quantities and counts is crucial to understanding the world. Evolutionary learning has given numerical cognition skills to several animals, including human beings (Dehaene, 2011). Our ancient ancestors furthered numeracy by developing multiple number systems, similar to but independent from the evolution of languages. Numeracy is an essential skill for language understanding, since numbers are often interspersed in text: the 6 million pages in English Wikipedia have over 150 million numbers.

Numbers are neglected. In NLP, however, numbers are either filtered out explicitly during preprocessing (Graff et al., 2003), or treated the same as words, often collapsing them into an `UNK` token. Subword tokenization approaches like BPE (Sennrich et al., 2016) and WordPiece (Wu et al., 2016) instead retain numbers, but split them into arbitrary tokens, for example `1234` might be split into two tokens as `12-34` or `123-4` or `1-234`.

Recent work has shown that these are suboptimal number representations (Wallace et al., 2019; Zhang et al., 2020). On the DROP Question Answering benchmark, BERT performs five times worse when the answer is a number instead of a span of text (Dua et al., 2019). Relatively simple strategies like switching from subword to char-level tokenization (Geva et al., 2020), or from decimal to scientific notation (Zhang et al., 2020) already boost performance. Such results warrant a deeper study into the best number representations.

Numbers are important. Given the ubiquity of numbers and their fundamental differences with words, enabling NLP systems to represent them effectively is beneficial for domains like scientific articles (Spithourakis and Riedel, 2018) and financial documents (Chen et al., 2019; Jiang et al., 2020). Number understanding is also useful to detect sarcasm (Dubey et al., 2019) and to model dialogues involving price negotiations (Chawla et al., 2020).

Recent NLP progress towards numeracy has been sporadic but encouraging. In this paper, we survey prior work and highlight the kind of numeracy targeted (e.g., arithmetic, measurement, numeration) as well as the kind of representation used (e.g., value embeddings, DigitRNNs). We provide the first NLP-centric taxonomy of numeracy tasks (Section 2) and of number representations (Section 3) for the reader to succinctly comprehend the challenge posed by numeracy. We synthesize key takeaways (Section 5) and propose a unifying vision for future research (Section 6).

	Benchmarking or Probing Tasks		Downstream Applications
	Abstract	**Grounded**	
Exact	Simple Arithmetic (2+3=5)	AWP (2 balls + 3 balls = 5 balls), Exact Facts (birds have two legs)	Question Answering, Science Problems
Approx	Numeration ('2' = 2.0), Magnitude ('2' < '5')	Measurement (dogs weigh 50 lbs), Numerical Language Modeling	Sarcasm Detection, Numeral Categorization

Table 1: Seven numeracy tasks, arranged along the axes of (rows) granularity - exact vs approximate, and (columns) units - abstract vs grounded. We also list downstream applications requiring a similar granularity of numeracy.

2 Tasks

There are several different aspects of numeracy. The DROP dataset alone offers a wide variety of numeric reasoning questions such as retrieval-based (*How many yards did Brady run?*), count-based (*How many goals were scored?* given a comprehension describing multiple goals), and simple arithmetic (*How many years after event 1 did event 2 occur?* given dates of both events). Besides downstream applications, there have also been probing experiments to evaluate whether NLP models can decode numbers from strings (e.g., *19* to `19.0`), or estimate quantities (e.g., *how tall are lions?*).

Such a diverse range of abilities are usually all referred to collectively as *numeracy*, which gives rise to confusion. We limit this abuse of terminology and provide a neat taxonomy for arranging the different tasks proposed under numeracy.

2.1 Our Taxonomy of Tasks

Drawing from work in cognitive science (Feigenson et al., 2004), we propose the following two dimensions to organize tasks within numeracy:

1. **Granularity**: whether the encoding of the number is (1) exact, e.g., *birds have* `two` *legs*, or (2) approximate, e.g., *Jon is about* `180` *cms tall*.

2. **Units**: whether the numbers are (1) abstract, e.g., 2+3=5, or (2) grounded, e.g., 2 apples + 3 apples = 5 apples. While abstract mathematical tasks are easy to probe and create artificial datasets for, numbers grounded in units are challenging since they need to be understood in the context of words.

2.2 Survey of Existing Tasks

We now describe 7 standard numeracy tasks, arranged according to our taxonomy in Table 1. We defer discussion on downstream tasks (rightmost column in the table) as well as miscellaneous numeracy-related tasks (such as counting) to A.

Simple Arithmetic is the task of addition, subtraction, etc. over numbers alone. It is convenient to create synthetic datasets involving such math operations for both masked (Geva et al., 2020) and causal language models (GPT-3 Brown et al. 2020).

Numeration or Decoding refers to the task of mapping a string form to its numeric value, e.g., *19* to `19.0`. Within NLP, this task is set up as a linear regressor probe over a (frozen) representation of the string. Numeration has been probed for in static word embeddings (Naik et al., 2019), contextualized language models (Wallace et al., 2019), and multilingual number words, e.g., *nineteen* or *dix-neuf* (Johnson et al., 2020).

Magnitude Comparison is the ability to tell which of two (or more) numbers is larger. For language models, this has been probed in an argmax setup (choose the largest of five numbers) as well as a binary classification task, e.g., given *23* and *32*, pick the label `1` to indicate that *32 > 23* (Naik et al., 2019; Wallace et al., 2019).

Arithmetic Word Problems (AWP) are the grounded version of simple arithmetic that we find in school textbooks, e.g., *Mary had two cookies. She gave one away. How many does she have left?* There exist several NLP datasets on math word problems (Amini et al., 2019; Saxton et al., 2019; Roy and Roth, 2015; Hendrycks et al., 2021).

Exact Facts in the context of numeracy involves commonsense knowledge such as *dice have* `6` *faces* or *birds have* `two` *legs*. An approximate sense of quantity would be of little help here since assertions like *dice have* `5` *faces* or *birds have* `three` *legs* are factually incorrect. Two recent datasets for numeric commonsense facts are Numbergame (Mishra et al., 2020) and NumerSense (Lin et al., 2020).

Measurement Estimation is a task in psychology in which subjects are asked to approximately guess measures of objects along certain dimensions, e.g., *number of seeds in a watermelon* or *weight*

of a telephone (Bullard et al., 2004). VerbPhysics (Forbes and Choi, 2017) is a benchmark of binary comparisons between physical attributes of various objects, e.g., ball $<_{size}$ tiger. DoQ (Elazar et al., 2019) is a web-extracted dataset of Distributions over Quantities, which can be used as a benchmark for language models' measurement estimation abilities (Zhang et al., 2020). Lastly, MC-TACO (Zhou et al., 2020) is a collection of temporal-specific measurement estimates, e.g., *going for a vacation spans a few days/weeks*.

Numerical Language Modeling in its literal sense is not a task but a setup, analogous to masked/causal language modeling for words. Other tasks could be modeled as numeric language modeling, e.g., arithmetic (*2+3=[MASK]*) and measurement estimation (*lions weigh [MASK] pounds*). In practice, numerical language modeling refers to the task of making numeric predictions for completing unlabelled, naturally occurring text.

Word predictions in language modeling are typically evaluated with classification metrics such as Accuracy at K or perplexity. Numeric predictions, on the other hand, are evaluated with regression metrics such as mean absolute error, root mean squared error, or their log-scaled and percentage variants.

Downstream Applications for numeracy are abound. Dubey et al. (2019) detect sarcasm in tweets based on numbers. Chen et al. (2020) identify claims in financial documents using alternative number representations and the auxiliary task of numeral understanding or categorization (Chen et al., 2018). Similarly, simple arithmetic and math word problems serve as auxiliary tasks for GenBERT (Geva et al., 2020) towards improving its score on the DROP QA benchmark.

3 Methods

Analogous to our taxonomy of subtasks in the previous section, here we attempt to arrange the wide variety of alternative number representations proposed in recent literature. We limit our analysis to methods of encoding (numbers $\rightarrow$ embeddings) and/or decoding (embeddings $\rightarrow$ numbers) numbers. We do not discuss, for example, methods that use symbolic reasoning (Andor et al., 2019) or modify activation functions to enhance numeracy (Trask et al., 2018).

A typical example of the base architecture could be BERT (Devlin et al., 2019), the workhorse of modern NLP. We assume that there exists an independent parallel process of mapping words into embeddings, such as subword tokenization followed by lookup embeddings in BERT.

3.1 Our Taxonomy

We look at two kinds of representations: string-based and real-based. Real-based representations perform some computation involving the numerical value of the number. The string-based representations instead see numbers in their surface forms; they must assign arbitrary token IDs and look up their embeddings to feed into the architecture.

3.1.1 String Based

By default, language models treat numbers as strings, the same as words. However, within string representations, one could tweak simple changes:

Notation: The number 80 could be written in Hindu-Arabic numerals (80), Roman numerals (LXXX), scientific notation (8e1), English words (eighty), or with base 20 as in French (quatre-vingts). Nogueira et al. (2021) exclusively study the effect of many such notation choices in language models, on the task of simple arithmetic.

Tokenization: Word level tokenizations are ineffective for numbers, since they are likely to map most numbers to an UNK token, except for a few commonly occuring ones (e.g., 1, 2, 5, 10, 100). Other possibilities are subword tokenizations like BPE and WordPiece, as well as character (or digit) level tokenizations.

Pooling: The pooling dimension of variation springs up after analyzing the effect of tokenization. With subword and character level tokenizations, a single number may now correspond to multiple tokens, e.g., 100 segmented into `10-0` or `1-0-0`. Prior work (Spithourakis and Riedel, 2018) has argued for using RNNs or CNNs to instead pool the embeddings of these tokens into a single embedding before feeding to the language model. The default way that language models see numbers are the same as words, hence no pooling is applied.

3.1.2 Real Based

Real-based number encoders can be expressed as $f : \mathbb{R} \rightarrow \mathbb{R}^d$ whereas decoders can be expressed as $g : \mathbb{R}^d \rightarrow \mathbb{R}$. Real-based methods proposed in literature can vary on account of direction (whether they encode, decode or both), scale (linear vs log), and discretization (binning vs continuous valued).

Direction: Some proposed methods are encoder-only, e.g., DICE (Sundararaman et al., 2020), while some can be decoder-only, e.g., those requiring sampling from a parameterized distribution (Berg-Kirkpatrick and Spokoyny, 2020).

Scale: Inspired by cognitive science literature (Dehaene, 2011), several methods have attempted to model numbers in the log (instead of linear) scale, i.e., to perform mathematical operations on the logarithm of the number to be represented. The first operation in a log-scaled f is $log(\cdot)$ and the last operation in a log-scaled g is $exp(\cdot)$. We discuss more scales in the following subsection, such as the stabilized log scale (Jiang et al., 2020) and the learned scale/flow (Berg-Kirkpatrick and Spokoyny, 2020).

Discretization: Training continuous value functions for a large range of numbers turns out to be practically infeasible (Wallace et al., 2019). Some real-based methods first bin numbers before learning embeddings for each bin. These bins could be on the linear scale (0-10, 10-20, 20-30, ...) or the log scale (0.01-0.1, 0.1-1, 1-10, ...), and the lookup embeddings can be learnt by the regular cross entropy (Chen et al., 2020) or dense cross entropy (Zhang et al., 2020).

3.2 Survey of Existing Methods

Having established dimensions of variance of number representations, we describe some key string-based and real-based methods used in prior work. Table 2 depicts these methods as individual rows, with the first three columns showing their position in our taxonomy (§ 3.1). The last seven columns correspond to the seven tasks (§ 2.2), with each cell denoting a representative work that introduce it.

3.2.1 String-based methods

Word Vectors & Contextualized Embeddings Word2vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014), ELMo (Peters et al., 2018), and BERT (Devlin et al., 2019) have been probed as baselines against several contending methods.

GenBERT Geva et al. (2020) presented GenBERT, a question answering model with pretrained BERT serving as both its encoder and decoder. GenBERT tokenizes numbers at the digit level, and is finetuned on auxiliary tasks of arithmetic word problems and simple arithmetic.

NumBERT Zhang et al. (2020) pretrain BERT from scratch over a modified dataset such that all numbers have been converted into scientific notation, i.e., 314.1 is expressed as 3141[EXP]2). NumBERT hence follows a scientific notation, subword tokenization, and no pooling.[1]

DigitRNN, DigitCNN Spithourakis and Riedel (2018) and Wallace et al. (2019) experimented with poolingof digit embeddings into a single embedding representing the full number. Both used RNNs as well as CNNs for pooling.

DigitRNN-sci & Exponent (Embedding) Berg-Kirkpatrick and Spokoyny (2020) used a scientific notation variant of DigitRNNs (which we refer to as DigitRNN-sci in Table 2), as well as a simpler alternative: exponent embedding. The latter merely learns a lookup embedding for the exponent, completely ignoring the mantissa.

3.2.2 Real-based methods

DICE Deterministic Independent-of-Corpus Embeddings (Sundararaman et al., 2020) is an attempt to handcraft number encoder [2] f so as to preserve the relative magnitude between two numerals and their embeddings. Given two scalars i and j, and their embeddings $f(i)$ and $f(j)$, the cosine distance between $f(i)$ and $f(j)$ is intended to monotonically increase/decrease with the Euclidean distance between i and j. DICE is offered as not only a deterministic encoding but also as an auxiliary loss function for softly training number embeddings alongside, say, SQuAD (Rajpurkar et al., 2016)

Value Embedding The most intuitive parameterized encoder for real numbers is one that feeds the scalar magnitude of the number through a shallow neural network. The converse of value embedding is to learn a shallow neural network mapping $g : \mathbb{R}^d \rightarrow \mathbb{R}$. This decoder is simply the probe used for decoding/numeration task.

The idea of projecting number magnitudes into an NLP model that otherwise inputs only lookup embeddings may appear flawed. But Vaswani et al. (2017) have (rather successfully) encoded positional information into transformers using both learned embeddings (similar to Value) and fixed ones (similar to DICE).

Log Value Wallace et al. (2019) also experiment with a log-scaled value encoder in addition to the one on a linear scale. Zhang et al. (2020) experiment with a log value decoder for measurement estimation, which they call the **RGR** (regress) method. Log scaling has a neuroscientific inspiration since

[1] Pooling as described in § 3.1.1.

[2] Number encoder-decoder as defined in § 3.1.2.

				Exact			Approximate			
				Arith	Facts	AWP	Num	Mag	Meas	LM^N
String-Based	**Notation**	**Tokenization**	**Pooling**							
Word Vectors	Decimal	Word	NA	W+19			W+19	W+19	G+19	J+20
Contextualized	Decimal	Subword	No	W+19	L+20	G+20	W+19	W+19	Z+20	SR18
GenBERT	Decimal	Char	No	G+20		G+20				
NumBERT	Scientific	Char	No						Z+20	Z+20
DigitRNN/CNN	Decimal	Char	Yes	W+19			W+19	W+19		SR18
DigitRNN-sci	Scientific	Char	RNN							BS20
Exponent	Scientific	Word	NA							BS20
Real-Based	**Scale**	**Direction**	**Binning**							
DICE	Linear	Enc-only	No	S+20			S+20	S+20		
Value	Linear	Both	No	W+19			W+19	W+19		
Log Value	Log	Both	No	W+19			W+19	W+19	Z+20	
MCC	Log	Dec-only	Yes						Z+20	
Log Laplace	Log	Dec-only	No							BS20
Flow Laplace	Learn	Dec-only	No							BS20
DExp	Log	Dec-only	No							BS20
GMM	Linear	Dec-only	Both**							SR18
GMM-proto	Linear	Enc-only*	No	J+20			J+20	J+20		J+20
SOM-proto	Log	Enc-only*	No	J+20			J+20	J+20		J+20

Table 2: An overview of numeracy in NLP: Each row is a method (§3.2), arranged as per our taxonomy (§3.1) split by string and real, further branching into three dimensions each. The last seven columns correspond to the seven subtasks of numeracy (§2.2), split by Exact and Approximate granularity (§2.1). The cells point to representative (not exhaustive) works that have experimented with a given method (row) on a given task (column). Notes: Prototype* is encoder-only but reuses embeddings for the decoder (Jiang et al., 2020). GMM** has been discretized (Spithourakis and Riedel, 2018) as well as continuous valued (Berg-Kirkpatrick and Spokoyny, 2020).

observations of human (and animal) understanding of numbers is better modelled by a log-scale representation (Dehaene, 2011).

Log Laplace In contrast to the point estimate output of the RGR decoder, models can also be used to parameterize a distribution over numbers. Such a formulation is helpful when estimating approximate quantities. Vectors representing some context can be used to parameterize, say, the mean and variance of a Gaussian or Laplace distribution. Berg-Kirkpatrick and Spokoyny (2020) instead transform the space being modeled by parameterizing the location parameter of a Log-Laplace distribution $L(X, 1)$ where X is the context representation of unmasked tokens, in a masked (numerical) language modelling setup. When inferring or decoding a number, they sample a point $z \sim L(X, 1)$ and exponentiate it, such that the output is $exp(z)$.

Flow Laplace The expressivity of number decoders can be expanded or contracted by merely parameterizing a different distribution. Berg-Kirkpatrick and Spokoyny (2020) propose a more expressive decoder where instead of the log scale, the model learns its own density mapping. After sampling $z \sim L(X, 1)$, the output is transformed to $\frac{exp(\frac{z-a}{b})}{c}$, where a, b, and c, are also parameters emitted by the same model.

MCC or multi-class classification is another number decoder which outputs a distribution, but a discrete one: over log-scaled bins of numbers, e.g., 1-10, 10-100, and so on (Zhang et al., 2020). Previously described decoders either output a point estimate or a unimodal distribution, thus failing to hedge its predictions for a multimodal ground truth. Given a masked number prediction problem *We went to the restaurant at [MASK] p.m.*, MCC is better equipped to estimate two peaks: one around lunch time (say, 1-2 p.m.) and another around dinner (say, 7-9 p.m.).

Discrete Latent Exponent (DExp) is another potentially multimodal distribution (Berg-Kirkpatrick and Spokoyny, 2020) where the model parameterizes a multinomial distribution for the exponent (similar to MCC) and uses it to sample an exponent e, which then acts as a latent variable for emitting the mean μ of a Gaussian (standard deviation fixed at 0.05). This Gaussian is finally used to sample the output number $z \sim N(\mu, 0.05)$.

GMM Another attempt to circumvent the unimodal Gaussians or point estimates is to learn a Gaussian mixture model. Spithourakis and Riedel (2018) learn a mixture of K Gaussians by pretraining their means (μ_i) and variances ($\sigma_i{}^2$) over the training corpus with Expectation Maximization

algorithms, while the mixing weights π_i are derived from the model. Next, to sample a single number from the GMM probability mass function $q(u) = \sum_{i=1}^{K} \pi_i N(u; \mu_i; \sigma_i)$, the authors first sample the precision (number of decimal places) from yet another Gaussian and use that to discretize the probability mass function into equal sized bins, over which the probabilities are summed. If the sampled precision is, say 2, then the probability of emitting a number 3.14 is given by $\int_{3.135}^{3.145} q(u)du$. This likelihood estimate is used to train a causal language model.

Berg-Kirkpatrick and Spokoyny (2020)'s GMM implementation is slightly different: it alters the last inference step by sampling directly from the mixture of Gaussians, as they did with Log Laplace, Flow Laplace, and DExp.

GMM-prototype by Jiang et al. (2020) similarly pretrains (with EM/hard-EM) the mean, the variances, but also the mixture weights π_is of a GMM over the training corpus. They then learn K prototype embeddings e_is corresponding to the K Gaussians. When encoding a new numeral n, its (input) embedding is calculated as: $E(n) = \sum_{i=1}^{K} w_i.e_i$, where the weights are induced from the GMM:

$$w_i = P(Z = i|U = n) = \frac{\pi_i N(n; \mu_i; \sigma_i)}{\sum_{j=1}^{K} \pi_j N(n; \mu_j; \sigma_j)}$$

Thus the difference between GMM and GMM-prototypes is that after fixing mean and standard deviations of the Gaussian mixtures, in GMM the model learns to predict the mixture weights π_i for each individual number prediction, whereas in GMM-prototype, π_i's are frozen and the model learns prototype embeddings e_i's. Note that prototype embeddings are encoder-only.To decode numbers, the authors implement weight-sharing across input and output embeddings, similar to how word vectors are trained (Mikolov et al., 2013), i.e., finding out which of the numerals in the corpus has the closest embedding.

SOM-prototype GMM-prototype, in effect, merely use the mixture of Gaussians to infer prototypes and to get the weights w_i's. Jiang et al. (2020) tried another variant by identifying prototype numerals with Self Organizing Maps (Kohonen, 1990) and by defining the weights as: $w_i = |g(x_i) - g(n)|^{-1}$ where x_i is the ith prototype, n is the number to be encoded, and g is a log-based squashing function.

4 Results

Having organized the landscape of numeracy tasks and methods, we now present come key results for each numeracy task in NLP from previously published experiments over a subset of the described number representations:

Abstract Probes Word Embeddings vastly outperform random embedding baselines on abstract probes such as numeration, magnitude comparison, and sorting (Wallace et al., 2019; Naik et al., 2019). DICE, Value and Log Value embeddings excel at these probes, which makes intuitive sense given that they explicitly encode the numbers' magnitude - although Value embeddings do not easily extrapolate to larger numbers, possibly due to instability in training. The best number encoders with respect to these probes were found to be DigitCNNs, and character-tokenized models, e.g., ELMo, in general outperform subword ones, e.g., BERT (Wallace et al., 2019).

Arithmetic GPT-3 (Brown et al., 2020) performs extremely well at zero shot simple arithmetic, as long as the number of digits in the operands are low. The tokenization scheme could be the cause for limited extrapolation, since language models get better at arithmetic when numbers are tokenized at the digit/character level (Nogueira et al., 2021; Wallace et al., 2019). For arithmetic word problems, state of the art solvers rely on predicting an equation, which is then filled in with specific numeric values from the question (Patel et al., 2021), altogether bypassing the need for encoding numbers into embeddings.

Masked Language Modelling Zhang et al. (2020) show that BERT pretrained over datasets where numbers are in scientific notation (NumBERT) converges to the same loss as BERT on masked language modelling objective, and scores nearly the same on GLUE language understanding benchmarks. For (causal) numeric language modelling, Spithourakis and Riedel (2018) show that Gaussian Mixture Models are the best decoders. For (masked) numeric language modelling, Berg-Kirkpatrick and Spokoyny (2020) show that modelling the mantissa in scientific notation may be an overkill, since exponent embeddings alone outperform DigitRNN-sci over financial news and scientific articles.

Measurement Estimation Zhang et al. (2020) train a regression probe to predict measurements of objects over the CLS embeddings of

BERT/NumBERT. Given a template-lexicalized sentence such as "the dog is heavy," the model must predict the weight of a typical dog, against ground truth from the Distribution over Quantities dataset (Elazar et al., 2019). They find that NumBERT is a better text encoder than BERT for measurement estimation, the only difference between them being the notation used by the respective pretraining corpora. They also experiment with two number decoders: MCC (multi-class classification) and RGR (regression / Log Value embedding). MCC performs better when trying to predict Distributions over Quantities - perhaps due to the ground truth resembling the predicted gaussians - but not on VerbPhysics - where the ground truth is less noisy. Lastly, even static word embeddings like GloVe have been shown to contain enough knowledge of measurement estimates to contrast two objects, e.g., classifying whether a *car* is bigger/heavier/fasster than a *ball* (Goel et al., 2019).

Exact Facts BERT and RoBERTa capture limited numerical commonsense, evident over NumerSense (Lin et al., 2020) sentences such as 'a tricycle has [MASK] wheels,' with the answer choices limited to the numbers 0-10. Results can be further improved by finetuning over a Wikipedia-extracted dataset of numeric information. Mishra et al. (2020) find the commonsense question answering to be one of the hardest among their Numbergame challenge, using the NumNetv2 model (Ran et al., 2019) which is commonly used for DROP question answering. Both of these experiments evaluate on exact match metrics, hence it remains to be seen if representing approximate magnitudes yields benefit in modelling numeric facts.

5 Recommendations

Based on the above results, we now synthesize certain key insights into a set of directed takeaways to guide practitioners' design of number representations for their task:

Rule of thumb for string-based methods? Scientific notation is superior to decimal notation (Zhang et al., 2020) since models can learn to attend mostly to the exponent embedding rather than the mantissa (Berg-Kirkpatrick and Spokoyny, 2020). Character level tokenization outperforms subword level (Nogueira et al., 2021; Wallace et al., 2019; Geva et al., 2020). Pooled representations (DigitRNN, DigitCNN) lack a controlled study with unpooled ones (NumBERT, GenBERT) which makes it hard to proclaim a winner among the two.

Rule of thumb for real-based methods? Log scale is preferred over linear scale (Zhang et al., 2020; Jiang et al., 2020; Wallace et al., 2019; Berg-Kirkpatrick and Spokoyny, 2020), which makes intuitive sense but lacks as rigorous a study as has been undertaken in the cognitive science community (Feigenson et al., 2004). Regarding discretization, Zhang et al. (2020) show that binning (dense cross entropy loss) works better than continuous value prediction (MAE loss) on datasets where ground truth distributions are available. Lastly, modeling continuous predictions is notoriously hard for large ranges (Wallace et al., 2019) but Spithourakis and Riedel (2018) offer a way of binning such distributions by picking a precision level.

Encoding vs Decoding numbers? In our simplified discussions above, we avoid differentiating between methods for encoding and decoding numbers. Value Embedding, for instance, can be used to encode numbers (projecting scalars onto vector space) as well as to decode numbers (collapsing a vector into a scalar). On the other hand, manually-designed encoders like DICE are not easily reversible into decoding methods. Even with reversible methods, the encoders and decoders must usually be independently parameterized, unlike the input and output word embeddings which often share weights (Press and Wolf, 2016). Prototype embeddings by Jiang et al. (2020) are an exception, which share input/output embeddings for a fixed vocabulary of numbers.

Can we mix-and-match multiple methods? Given the wide range of number representations, an obvious next step is to try an ensemble of embeddings. Berg-Kirkpatrick and Spokoyny (2020) show that for encoding numbers, exponent embeddings added to DigitRNN (scientific notation) embeddings barely outperforms the exponent embeddings alone. Similar experiments with a mix of real and string methods are yet to be seen.

Which methods for which tasks? Based on our taxonomy of tasks in Table 1, abstract tasks are good early probes for the grounded ones, e.g., finetuning GenBERT (Geva et al., 2020) on simple arithmetic helps it do well on downstream question answering, and the high scores of DICE (Sundararaman et al., 2020) on numeration and magnitude comparison are an indicator of similar boosts on (numeric) language modelling. With respect

to granularity, real-based methods work well for approximate tasks such as measurement estimation and language modeling (Zhang et al., 2020; Berg-Kirkpatrick and Spokoyny, 2020) but not for exact tasks like arithmetic word problems or commonsense. DigitRNNs are broad-purpose number encoders, whereas distribution modeling methods like DExp are effective at decoding numbers.

6 Vision for Unified Numeracy in NLP

Numeracy is a core system of human intelligence (Kinzler and Spelke, 2007). Teaching numeracy to students works best when taught holistically, while less effective teachers deal with areas of mathematics discretely (Askew and Askew, 1997). While the NLP community has genuinely strived to improve language models' numeric skills, not all aspects of numeracy have been sufficiently targeted. It is evident from the sparsity in Table 2 that the community is far from achieving, or attempting, a holistic solution to numeracy. In this section, we outline our vision for such a unified solution, as three prerequisites necessary to consider for numerical NLU:

Evaluation. The first step towards a holistic solution to numeracy requires a benchmark covering its different subtasks. Aggregated leaderboards in NLP like GLUE (Wang et al., 2018) and SuperGLUE (Wang et al., 2019) have incentivized research on natural langauge understanding, with scores categorized into semantic, syntactic, logical, and background knowledge.

An analogous leaderboard could be constructed to evaluate models on numeric reasoning tasks, again categorized according to the skills evaluated, e.g., exact vs approximate granularity, or abstract vs grounded numeracy. Numbergame (Mishra et al., 2020) is one such aggregation focusing on exact numeracy benchmarks, as evaluated by F1 and exact match scores in a reading comprehension setup. Both Numbergame and our own list of tasks (Section 2.2) are preliminary attempts at teasing apart the different aspects of numeracy. We encourage researchers to extend and refine such taxonomies.

A suite of numeracy tasks, matched with evaluations of their respective numerical skills, can enable testing model generalization from one skill to another. Some progress has already been made in this transfer learning setup, e.g., GenBERT (Geva et al., 2020), finetuned on a synthetic dataset of arithmetic problems, is found to score higher on DROP QA. Similarly, DICE (Sundararaman et al., 2020), optimized for numeration, improves score on Numeracy600K order-of-magnitude prediction task. Going forward, we need several such studies, ideally for each pair of tasks to see whether some numeracy skills help models generalize to others.

Design Principles. Number representations vary based on design trade-offs between inductive biases and data-driven variance. The default BERT setup, with subword tokenization and lookup embeddings, occupies the variance end of the spectrum, allowing freedom in representing numbers. Value embeddings and DICE encodings, on the other hand, are closer to the bias end of the spectrum, since the inductive bias of continuity on the number line constrains the learning space. It is important to identify where on the bias-variance scale any representation stands, for a fair comparison.

Following parallel work in cognitive science, the community could explore whether exact and approximate numeracy require two specialized modules (Feigenson et al., 2004) or could be handled with a single representation (Cordes et al., 2001).

Model designers must also make a choice on coverage: whether to target a broad or a narrow range of numbers to be represented. Multi-class classification (Zhang et al., 2020) over a fixed number of bins, restricts the range of numbers expressed, as do DICE embeddings (Sundararaman et al., 2020). Value embeddings are continuous and theoretically unrestricted, but must practically be capped for bug-free training. On the other hand, string-based representations could always fall back to subword/char-level token embeddings to represent not only floats but also irrational ($\sqrt{2}$) and complex ($1 + 2\iota$) numbers. Roy et al. (2015) introduced the Quantity-Value Representation format to allow closed and open ranges alongside scalar point numbers.

Broader Impact. Numbers are ubiquitous in natural language and are easily identified, at least in numeral forms. But they are by no means the only class of ordered concepts required for natural language understanding. Successful number representations can inspire work on incorporating more continuous domains into natural language processing systems. For instance, gradable adjectives like good, great, amazing, etc. are arguably on some cardinal scale, which can be mapped using value embeddings or Gaussian mixture models (Sharp et al., 2018; de Marneffe et al., 2010). Days of the

week (Mon-Sun) and months of an year (Jan-Dec) form periodic patterns which can be modeled with sinusoidal functions (Martinez et al., 2020).

Lastly, numeracy is essential for natural language understanding. Consider the sentence: "*Programmers earn $200,000 versus $100,000 for researchers.*" An intelligent agent with numeracy skills would identify that $100k is half of $200k, that $100k possibly denotes annual salary, and infer that higher salaries lead to higher standards of living. In short, it was able to learn something about the two concepts *programmers* and *researchers*, by crossing the continuous semantic space of numbers! The agent could now make use of this knowledge in a number-free situation, e.g., the mask in "*He could not afford a car for several years after earning a CS degree because she took a job as a [MASK]*" might better be filled with the word *researcher*, than with *programmer*. A key goal of imparting numeracy to NLP models is to help them understand more about the world, *using* numbers.

7 Conclusion

This paper summarizes and contextualizes recent work on numeracy in NLP. We propose the first taxonomy of tasks and methods concerning text-centric numeric reasoning. We highlight key takeaways from the several experiments in literature, along with caveats and scope for confirming some of the observed trends. We present a case for lack of a holistic solution to numeracy in NLP, and put forward a set of aspects to consider when working towards one. We draw the following two major conclusions from our study: (1) the default subword segmentation with lookup embeddings used to represent words is clearly suboptimal for numbers (2) there are several unanswered research questions on the level of specificity, coverage, and inductive bias needed to holistically solve numeracy.

8 Acknowledgements

This work was funded by the Defense Advanced Research Projects Agency with award N660011924033. We would like to thank the countless suggestions we accumulated during preliminary presentations at MLSS 2020, WeCNLP 2020, and GSS 2020, as well as over email correspondences with Biplav Srivastava, Antoine Bosselut, and Harsh Agarwal. We would like to thank the anonymous NAACL 2021 reviewers (particularly #3) for pointing out blind spots in our submission, which we have tried our best to rectify.

Ethical Considerations

This work revolves around the Hindu-Arabic Numeral system and English number words, which are not the only number systems still in use today. We encourage follow-up work to take these systems into consideration, on the lines of Johnson et al. (2020) and Nefedov (2020).

References

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2357–2367, Minneapolis, Minnesota. Association for Computational Linguistics.

Daniel Andor, Luheng He, Kenton Lee, and Emily Pitler. 2019. Giving BERT a calculator: Finding operations and arguments with reading comprehension. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5947–5952, Hong Kong, China. Association for Computational Linguistics.

Mike Askew and Mike Askew. 1997. *Effective teachers of numeracy*. King's College London London.

Taylor Berg-Kirkpatrick and Daniel Spokoyny. 2020. An empirical investigation of contextualized number prediction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4754–4764, Online. Association for Computational Linguistics.

Satwik Bhattamishra, Arkil Patel, and Navin Goyal. 2020. On the computational power of transformers and its implications in sequence modeling. In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 455–475, Online. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Sarah E Bullard, Deborah Fein, Mary Kay Gleeson, Nita Tischer, Robert L Mapou, and Edith Kaplan. 2004. The biber cognitive estimation test. *Archives of clinical neuropsychology*, 19(6):835–846.

Kushal Chawla, Gale Lucas, Jonathan Gratch, and Jonathan May. 2020. Bert in negotiations: Early prediction of buyer-seller negotiation outcomes. *arXiv preprint arXiv:2004.02363*.

Chung-Chi Chen, Hen-Hsen Huang, and Hsin-Hsi Chen. 2020. Numclaim: Investor's fine-grained claim detection. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*, CIKM '20, page 1973–1976, New York, NY, USA. Association for Computing Machinery.

Chung-Chi Chen, Hen-Hsen Huang, Yow-Ting Shiue, and Hsin-Hsi Chen. 2018. Numeral understanding in financial tweets for fine-grained crowd-based forecasting. In *2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, pages 136–143. IEEE.

Chung-Chi Chen, Hen-Hsen Huang, Hiroya Takamura, and Hsin-Hsi Chen. 2019. Numeracy-600K: Learning numeracy for detecting exaggerated information in market comments. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6307–6313, Florence, Italy. Association for Computational Linguistics.

Peter Clark, Oren Etzioni, Daniel Khashabi, Tushar Khot, Bhavana Dalvi Mishra, Kyle Richardson, Ashish Sabharwal, Carissa Schoenick, Oyvind Tafjord, Niket Tandon, et al. 2019. From'f'to'a'on the ny regents science exams: An overview of the aristo project. *arXiv preprint arXiv:1909.01958*.

Sara Cordes, Rochel Gelman, Charles R Gallistel, and John Whalen. 2001. Variability signatures distinguish verbal from nonverbal counting for both large and small numbers. *Psychonomic bulletin & review*, 8(4):698–707.

Marie-Catherine de Marneffe, Christopher D. Manning, and Christopher Potts. 2010. "was it good? it was provocative." learning the meaning of scalar adjectives. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 167–176, Uppsala, Sweden. Association for Computational Linguistics.

Stanislas Dehaene. 2011. *The number sense: How the mind creates mathematics*. OUP USA.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Abhijeet Dubey, Lakshya Kumar, Arpan Somani, Aditya Joshi, and Pushpak Bhattacharyya. 2019. "when numbers matter!": Detecting sarcasm in numerical portions of text. In *Proceedings of the Tenth Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 72–80, Minneapolis, USA. Association for Computational Linguistics.

Yanai Elazar and Yoav Goldberg. 2019. Where's my head? Definition, data set, and models for numeric fused-head identification and resolution. *Transactions of the Association for Computational Linguistics*, 7:519–535.

Yanai Elazar, Abhijit Mahabal, Deepak Ramachandran, Tania Bedrax-Weiss, and Dan Roth. 2019. How large are lions? inducing distributions over quantitative attributes. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3973–3983, Florence, Italy. Association for Computational Linguistics.

Lisa Feigenson, Stanislas Dehaene, and Elizabeth Spelke. 2004. Core systems of number. *Trends in cognitive sciences*, 8(7):307–314.

Maxwell Forbes and Yejin Choi. 2017. Verb physics: Relative physical knowledge of actions and objects. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 266–276, Vancouver, Canada. Association for Computational Linguistics.

Mor Geva, Ankit Gupta, and Jonathan Berant. 2020. Injecting numerical reasoning skills into language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 946–958, Online. Association for Computational Linguistics.

Pranav Goel, Shi Feng, and Jordan Boyd-Graber. 2019. How pre-trained word representations capture commonsense physical comparisons. In *Proceedings of the First Workshop on Commonsense Inference in Natural Language Processing*, pages 130–135, Hong Kong, China. Association for Computational Linguistics.

David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2003. English gigaword. *Linguistic Data Consortium, Philadelphia*, 4(1):34.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

Chengyue Jiang, Zhonglin Nian, Kaihao Guo, Shanbo Chu, Yinggong Zhao, Libin Shen, and Kewei Tu. 2020. Learning numeral embedding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2586–2599, Online. Association for Computational Linguistics.

Devin Johnson, Denise Mak, Andrew Barker, and Lexi Loessberg-Zahl. 2020. Probing for multilingual numerical understanding in transformer-based language models. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 184–192, Online. Association for Computational Linguistics.

Katherine D Kinzler and Elizabeth S Spelke. 2007. Core systems in human cognition. *Progress in brain research*, 164:257–264.

Teuvo Kohonen. 1990. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480.

Bill Yuchen Lin, Seyeon Lee, Rahul Khanna, and Xiang Ren. 2020. Birds have four legs?! NumerSense: Probing Numerical Commonsense Knowledge of Pre-Trained Language Models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6862–6868, Online. Association for Computational Linguistics.

Richard Diehl Martinez, Scott Novotney, Ivan Bulyko, Ariya Rastrow, and Andreas Stolcke. 2020. Contextual datetime language model adaptation for speech recognition. *West Coast NLP Summit*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Swaroop Mishra, Arindam Mitra, Neeraj Varshney, Bhavdeep Sachdeva, and Chitta Baral. 2020. Towards question format independent numerical reasoning: A set of prerequisite tasks.

Aakanksha Naik, Abhilasha Ravichander, Carolyn Rose, and Eduard Hovy. 2019. Exploring numeracy in word embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3374–3380, Florence, Italy. Association for Computational Linguistics.

Mikhail Nefedov. 2020. Dataset for evaluation of mathematical reasoning abilities in russian. In *Conference on Artificial Intelligence and Natural Language*, pages 135–144. Springer.

Rodrigo Nogueira, Zhiying Jiang, and Jimmy Li. 2021. Investigating the limitations of the transformers with simple arithmetic tasks. *arXiv preprint arXiv:2102.13019*.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are nlp models really able to solve simple math word problems?

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Marten Postma, Filip Ilievski, and Piek Vossen. 2018. SemEval-2018 task 5: Counting events and participants in the long tail. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 70–80, New Orleans, Louisiana. Association for Computational Linguistics.

Ofir Press and Lior Wolf. 2016. Using the output embedding to improve language models. *arXiv preprint arXiv:1608.05859*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Qiu Ran, Yankai Lin, Peng Li, Jie Zhou, and Zhiyuan Liu. 2019. NumNet: Machine reading comprehension with numerical reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2474–2484, Hong Kong, China. Association for Computational Linguistics.

Abhilasha Ravichander, Aakanksha Naik, Carolyn Rose, and Eduard Hovy. 2019. EQUATE: A benchmark evaluation framework for quantitative reasoning in natural language inference. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 349–361, Hong Kong, China. Association for Computational Linguistics.

Subhro Roy and Dan Roth. 2015. Solving general arithmetic word problems. In *Proceedings of the 2015*

Conference on Empirical Methods in Natural Language Processing, pages 1743–1752, Lisbon, Portugal. Association for Computational Linguistics.

Subhro Roy, Tim Vieira, and Dan Roth. 2015. Reasoning about quantities in natural language. *Transactions of the Association for Computational Linguistics*, 3:1–13.

David Saxton, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. 2019. Analysing mathematical reasoning abilities of neural models. In *International Conference on Learning Representations*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Rebecca Sharp, Mithun Paul, Ajay Nagesh, Dane Bell, and Mihai Surdeanu. 2018. Grounding gradable adjectives through crowdsourcing. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Georgios P. Spithourakis and Sebastian Riedel. 2018. Numeracy for language models: Evaluating and improving their ability to predict numbers. *CoRR*, abs/1805.08154.

Dhanasekar Sundararaman, Shijing Si, Vivek Subramanian, Guoyin Wang, Devamanyu Hazarika, and Lawrence Carin. 2020. Methods for numeracy-preserving word embeddings. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4742–4753, Online. Association for Computational Linguistics.

Mirac Suzgun, Yonatan Belinkov, Stuart Shieber, and Sebastian Gehrmann. 2019. LSTM networks can perform dynamic counting. In *Proceedings of the Workshop on Deep Learning and Formal Languages: Building Bridges*, pages 44–54, Florence. Association for Computational Linguistics.

Oyvind Tafjord, Peter Clark, Matt Gardner, Wen-tau Yih, and Ashish Sabharwal. 2019. Quarel: A dataset and models for answering questions about qualitative relationships. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7063–7071.

Alon Talmor, Oyvind Tafjord, Peter Clark, Yoav Goldberg, and Jonathan Berant. 2020. Leap-of-thought: Teaching pre-trained models to systematically reason over implicit knowledge. *Advances in Neural Information Processing Systems*, 33.

Alberto Testolin, Serena Dolfi, Mathijs Rochus, and Marco Zorzi. 2020. Visual sense of number vs. sense of magnitude in humans and machines. *Scientific reports*, 10(1):1–13.

Andrew Trask, Felix Hill, Scott E Reed, Jack Rae, Chris Dyer, and Phil Blunsom. 2018. Neural arithmetic logic units. In *Advances in Neural Information Processing Systems*, pages 8035–8044.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Eric Wallace, Yizhong Wang, Sujian Li, Sameer Singh, and Matt Gardner. 2019. Do NLP models know numbers? probing numeracy in embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5307–5315, Hong Kong, China. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, pages 3266–3280.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Sandra Williams and Richard Power. 2010. A fact-aligned corpus of numerical expressions. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation.

Karen Wynn. 1990. Children's understanding of counting. *Cognition*, 36(2):155–193.

Xikun Zhang, Deepak Ramachandran, Ian Tenney, Yanai Elazar, and Dan Roth. 2020. Do language embeddings capture scales? In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4889–4896, Online. Association for Computational Linguistics.

Ben Zhou, Qiang Ning, Daniel Khashabi, and Dan Roth. 2020. Temporal common sense acquisition with minimal supervision. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7579–7589, Online. Association for Computational Linguistics.

A Other Numeracy Tasks

Here, we describe certain related tasks that fall outside our taxonomy:

(Numeric) Paraphrasing is what we call the task of identifying one-to-one correspondences between different surface forms of the same number. Twelve is the same as '12', also referred to as a dozen. This task cuts across all the tasks we discussed, since the same number, expressed in several different ways, should be nevertheless identified by an NLP model before any subsequent reasoning. Similar to how WordNet (Miller, 1995) provides a huge list of synonyms, numeric paraphrases can be obtained by libraries[3] which convert numerals to words, words to numerals, etc. One could also envision this as a learning task given a large enough corpus, such as the NumGen dataset (Williams and Power, 2010) containing 2000 fact-aligned numeric expressions over 110 articles.

Quantity Entailment tasks (Ravichander et al., 2019; Roy et al., 2015) are analogous to Natural Language Inference, which requires understanding of not only equivalence (as in paraphrasing) but also deeper relations like entailment and contradiction, e.g., the premise 'he was 16 yrs old' entails the hypothesis 'he was a teenager'. On similar lines, Mishra et al. (2020) modified the existing QuaRel dataset (Tafjord et al., 2019) to force models to perform quantity entailment, e.g., *dog1 is light, dog2 is heavy* is replaced with *dog1 weighs 70 lbs, dog2 weighs 90 lbs*.

Numeral Understanding is the task of categorizing numbers into percentages, prices, dates, times, quantities, etc. and their respective subcategories (Chen et al., 2018).

Fused-Head Resolution for numbers is essential to ground them when the context is implicit. For example, the sentence "*I woke up at 11*" has 'a.m.' or 'o'clock' as the fused head to be resolved (Elazar and Goldberg, 2019).

Counting is the task of keeping track of discrete instances of some object. When kids count a set of objects, they quickly learn to keep a track, say on their fingers, but struggle with realizing the Cardinal Principle, i.e., the last counter value denotes the number of entities being considered (Wynn, 1990). Similarly, LSTMs (Suzgun et al., 2019) and transformers (Bhattamishra et al., 2020) have been shown to possess counting skills but in order to answer counting questions, they must also learn to map the counts to number words or numerals. Counting tasks have been proposed in computer vision (Testolin et al., 2020) as well as in NLP (Postma et al., 2018; Talmor et al., 2020).

Domain-specific tasks require background knowledge in addition to mathematical skills. Numbergame (Mishra et al., 2020) includes questions on Physics (*find the distance travelled in 2 hrs by a train moving at 50 mph*) and Chemistry (*find the mass percentage of H in C6H6*). Project Aristo (Clark et al., 2019) solves elementary and high school science problems, which often involve numeric reasoning.

[3]Example: https://pypi.org/project/num2words/