*Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022)*, pages 351–362
Marseille, 20-25 June 2022


# ArMATH: a Dataset for Solving Arabic Math Word Problems

**Reem Alghamdi[1], Zhenwen Liang[2], Xiangliang Zhang[12]**
[1]King Abdullah University of Science and Technology (KAUST)
[2]University of Notre Dame
reem.alghamdi@kaust.edu.sa, {zliang6, xzhang33}@nd.edu

**Abstract**

This paper studies solving Arabic Math Word Problems by deep learning. A Math Word Problem (MWP) is a text description of a mathematical problem that can be solved by deriving a math equation to reach the answer. Effective models have been developed for solving MWPs in English and Chinese. However, Arabic MWPs are rarely studied. This paper contributes the first large-scale dataset for Arabic MWPs, which contains 6,000 samples of primary-school math problems, written in Modern Standard Arabic (MSA). Arabic MWP solvers are then built with deep learning models and evaluated on this dataset. In addition, a transfer learning model is built to let the high-resource Chinese MWP solver promote the performance of the low-resource Arabic MWP solver. This work is the first to use deep learning methods to solve Arabic MWP and the first to use transfer learning to solve MWP across different languages. The transfer learning enhanced solver has an accuracy of 74.15%, which is 3% higher than the solver without using transfer learning. We make the dataset and solvers available in public for encouraging more research of Arabic MWPs: `https://github.com/reem-codes/ArMATH`.

**Keywords:** Math word problem, Transfer learning, Arabic math word problem

## 1. Introduction

Math Word Problems (MWP) are short paragraphs describing a mathematical problem and asking for an unknown quantity. *Table 1* shows an example of an MWP. These problems vary in difficulty with respect to the solution and language used to convey the problems. For example, these problems could be covering advanced mathematical concepts such as calculus or be written in a certain format like GRE questions. With the success of deep learning in Natural Language Processing, MWP has been recently studied for neuro-symbolic reasoning with deep neural networks (Amini et al., 2019a; Miao et al., 2020; Qin et al., 2020; Wang et al., 2018; Wang et al., 2019b; Liu et al., 2019; Xie and Sun, 2019a; Wang et al., 2017a; Li et al., 2019a; Zhang et al., 2020b; Wu et al., 2020; Liang and Zhang, 2021).

| | |
|---|---|
| **Question (English)** | Ali ate 3 apples from the basket and his brother ate 2. If the basket originally had 8 apples, how many are left? |
| **Question (Arabic)** | أكل علي 3 تفاحات من سلة فواكه، وتناول أخوه تفاحتين؛ فكم تفاحة تبقى في السلة إذا كان فيها 8 تفاحات في البداية؟ |
| **Equation** | $x = 8 - 3 - 2$ |
| **Answer** | 3 |

Table 1: An example of an MWP

All the prior MWP solvers are developed on Chinese and English MWP datasets. Although deep-learning-based models can be language-agnostic and translation systems can be helpful, it is essential to build an Arabic MWP solver trained by Arabic MWPs. There are several motivations. First, using machine translation may not be precise, which will lead to less-than-optimal results. Second, even if translated by humans, direct translation does not consider the cultural difference between the source language and Arabic. For instance, Arabic names, famous cities, food, plants, holidays, and geography will almost always appear in the question body. It is therefore essential for MWP solvers to be able to understand these differences.

In this paper, the first large-scale Arabic MWP dataset is created, which contains 6000 samples of primary-school math problems written in Modern Standard Arabic (MSA). Arabic MWP solvers are then built with deep learning models and verified on this dataset for their effectiveness. In addition, a transfer learning model is built to let the high-resource Chinese MWP solver promote the performance of the low-resource Arabic MWP solver. This work is the first to use deep learning methods to solve Arabic MWP and the first to use transfer learning to solve MWP across different languages. We make the dataset and solvers available in public for encouraging more research of Arabic MWPs: `https://github.com/reem-codes/ArMATH`.

## 2. Background

### 2.1. MWP Datasets

MWP solving is a special natural language understanding task. The related and previously used datasets differ in multiple ways:

- **Natural language of the problem description**: The most obvious difference is the natural language used to write the math problems. The most popularly used datasets are MAWPS dataset (Koncel-Kedziorski et al., 2016) in English and MATH23K dataset (Wang et al., 2017b) in Chinese.
- **Question types**: MWPs can be asked as one or more of the following types: find the answer, explain the steps, resonate, or choose the correct answer and justify. The most popularly studied MWPs cover one type of question. For instance, MathQA dataset (Amini et al., 2019b) questions are all multiple choices, while MAWPS and MATH23K cover questions asking to find a numerical value.
- **Difficulty levels**: There are datasets covering grade-school/general-level math topics like algebra or basic geometry. For instance, MATH23K and MAWPS datasets are for primary school math. Other datasets cover advanced or specific math (e.g., calculus and statistics) (Saxton et al., 2019). There are also datasets even covering specific tests questions, such as GRE like MathQA dataset (Amini et al., 2019b).

Datasets and benchmarks are crucial for developing machine learning methods in one research field. Regarding the publicly available MWP datasets, the most famous ones are the Chinese MATH23K (Wang et al., 2017b) and the English MAWPS (Koncel-Kedziorski et al., 2016). Both datasets contain single variable primary-school MWPs. Each MWP has a solution equation and a numerical answer, as shown in *Table 1.* There are 23,160 samples in MATH23K and 3,320 samples in MAWPS, on which a group of solvers has been developed (Xie and Sun, 2019a; Wu et al., 2020; Shen and Jin, 2020; Zhang et al., 2020b; Liang and Zhang, 2021). Our contributed Arabic MWP dataset includes single variable primary-school MWPs, similar to MATH23K and MAWPS.

### 2.2. Related Works for Solving MWPs

The development of MWP solving methods can be roughly divided into two stages. Earlier studies (Hosseini et al., 2014; Mitra and Baral, 2016) have attempted to introduce statistical machine learning methods to deal with MWP. Some researchers (Shi et al., 2015; Huang et al., 2017; Liang et al., 2018; Zou and Lu, 2019) found that the semantic parsing method is suitable for discovering effective features which is beneficial for the generation of solutions. However, these methods are non-scalable and lack generalizability as tremendous works are needed to design effective features and templates.

In recent years, deep learning methods have become dominant in this area. (Wang et al., 2017a) first proposed to apply sequence-to-sequence (Seq2Seq) framework to solve MWP and achieved satisfactory performance compared with previous methods. On the one hand, most following works focused on the generation module, i.e., the decoder. (Wang et al., 2019b) proposed a two-stage method to decompose the goals into two parts. (Liu et al., 2019; Xie and Sun, 2019a) proposed to use a tree structure decoder. (Chiang and Chen, 2018) introduced a stack-related decoder. Multiple decoder architecture (Zhang et al., 2020a; Shen and Jin, 2020) was also introduced to improve generation results. On the other hand, several works (Li et al., 2019a; Wang et al., 2018) focused on improving the encoding component. (Zhang et al., 2020b; Shen and Jin, 2020) chose to model quantity information with a sequential combination of RNN and GNN encoder. In (Liang and Zhang, 2021), a teacher module is proposed to make the encoder generate the representation matching the correct solution but disaccording to the wrong solutions.

With our Arabic MWP dataset and other publicly available MWP datasets, we study for the first time to use transfer learning to promote the performance of low-resource Arabic MWP solver based on the high-resource solvers.

### 2.3. Arabic Math Word Problems

Despite the rapid research progress in Chinese and English MWP solving, there are hardly any datasets for other languages, Arabic included. For Arabic MWP solving, only one paper was published (Siyam et al., 2017). This paper used statistical approaches to tackle the problem instead of applying deep learning methods, and the dataset used is a translation of 500 samples from an English dataset, instead of one reflecting Arabic culture.

## 3. Arabic MWP Dataset Creation

Arabic MWPs collection is a non-trivial task. The Chinese MATH23K dataset was curated by web crawling plenty of elementary-school, one-unknown-variable linear math word problems, then cleaning them, and finally adding equations to them (Wang et al., 2017b). MAWPS was made by extending on multiple smaller, previously studied English datasets, then crawling websites and adding some more (Koncel-Kedziorski et al., 2016).

| Tag | Equation | Segmented | Question |
|---|---|---|---|
| novel | $x = 23 * 2$ | اصطاد علي 23 سمكه واصطاد سعود مثليها فما عدد ما اصطاده سعود ؟ | اصطاد علي 23 سمكة واصطاد سعود مثليها، فما عدد ما اصطاده سعود؟ |
| novel | $x = 18 + 8$ | كم عصفورا كان علي الشجره اذا علمت انه بعد ان طار منهم 8 بقي 18 عصفورا ؟ | كم عصفورًا كان على الشجرة، إذا علمت أنه بعد أن طار منهم 8 بقي 18 عصفورًا؟ |
| inspired | $x = 6 * (2/5)$ | كم كيلومتر تساوي (2/5) من 6 كيلومترات = | كم كيلومتر تساوي ( 2/5 ) من 6 كيلومترات = |
| inspired | $x = 160 * 35$ | زرع غسان 35 صفا من الازهار في كل صف 160 زهره ؛ فكم زهره زرع ؟ | زرع غسان 35 صفًا من الأزهار، في كل صف 160 زهرة؛ فكم زهرة زرع؟ |
| inspired | $x = 50 * 40\%$ | 40% من 50 = . | 40 ٪ من 50 =. |
| novel | $x = 188/3.14$ | ما هو طول القطر للدائره التي يساوي محيطها 188 سم ؟ | ما هو طول القطر للدائرة التي يساوي محيطها 188 سم؟ |
| novel | $x = 876 - 343$ | اوجد قيمه : 876 - 343 = | أوجد قيمة: 876-343= |
| inspired | $x = 1/8$ | ما هو النظير الضربي ل 8 ؟ | ما هو النظير الضربي ل 8؟ |
| inspired | $x = 36/9$ | 36 شجره مزروعه في 9 صفوف ؛ فكم عدد الاشجار في كل صف ؟ | 36 شجرة مزروعة في 9 صفوف؛ فكم عدد الأشجار في كل صف؟ |
| novel | $x = 677 - 563$ | اوجد ناتج : 677 - 563 = | أوجد ناتج: 677-563= |

Table 2: Samples from the constructed ArMATH dataset

These approaches assume that web crawling is possible due to data availability in some websites containing questions. Alternatively, smaller datasets exist and can be cleaned. Both assumptions do not hold for Arabic. In terms of Arabic content, no such websites exist for large-scale web crawling. The Arabic dataset has to be created from scratch.

This paper introduces **ArMATH**: an Arabic single variable primary-school MWP dataset written in Modern Standard Arabic (MSA). In the first stage of constructing this dataset, five creative writers who are native Arabic speakers were hired to write question-equation pairs resembling the official Saudi primary-school maths books questions offered by The Ministry of Education[1], ensuring that the questions reflect primary-school level math and the Arabic people's names, date system, food, and cultural events. Mining the questions from the books only was not feasible for multiple reasons; first, many questions were not single-variable, single-answer questions. Second, some included a graph or a figure along the question. Third, although some questions were taken directly from the books, they were not enough to make a large-scale dataset. Therefore, creating more questions similar to the one officially offered was needed. After that, two professional translation agencies worked on translating some MATH23K samples from Chinese to Arabic. The translation was not literal but instead captured the idea behind the question, then altered it to fit the Arab naming as explained above. For instance, Chinese names were changed to Arabic names, and some fruit names that are not common in the Arabic world were changed. In addition, some of the translated questions were drastically changed, resulting in a change in the equation. It is better to think of these samples as *inspired* by MATH23K, rather than *translated from* MATH23K. Finally, professionals proofread the questions to ensure grammatical correctness.

To ensure integrity, the question-equation pairs were verified by a native Arabic speaker and a Saudi primary school math teacher to check their correctness. Then, a script ran through the MWP instance-equation pairs to check whether numbers appearing in the equation appeared in the MWP instance. If a number in the equation does not appear in the instance, we check whether it is a constant. The script eliminated some hard-to-spot spelling mistakes, such as writing "122" by mistake as "112".

The dataset has many different writing styles for the questions, sometimes repeating the exact phrase but asking for a different quantity, which is desirable. As discussed in a new paper (Patel et al., 2021), introducing variations on questions is essential to ensure the model does not treat MWP as bag-of-words only but also try to understand the relationship between words.

*Table 2* shows samples from the dataset. The dataset can be found at: `https://github.com/reem-codes/ArMATH`. Each sample contains the question-equation pair and tag information of the source: *novel* for creatively written samples and *inspired* for MATH23K-inspired samples. In *Table 2*, *segmented* is the preprocessed question for faster use. Preprocessing was done as follows:

- Indian numbers were converted to Arabic numbers
- Numeric words were converted into numbers: stand-alone numeric words, such as أربع تفاحات, are converted accordingly. However, they are

[1] `https://moe.gov.sa`

not detected when it is a part of the word, such as تفاحتين

- power sign was uniformly defined as ^
- Arabic special characters were mapped to English ones or eliminated.
- Arabic Tashkeel and madd were removed
- Ha'a and Taa were normalized
- Hamza forms were normalized
- Special characters (e.g., a tab) were converted accordingly.
- Spacing: adding spaces between numbers, operations, words, and punctuation for tokenization. In addition to spacing fraction/percentages correctly to be detected easily.

## 4. ArMATH Dataset Analysis

ArMATH dataset contains 6,000 samples (question-equation pairs): 3,533 samples are from creative writing, and 2,467 MATH23K-inspired samples. The ArMATH dataset was randomly split into 5 folds, 1,200 samples in each, for future 5-fold cross-validation if needed.

The dataset covers one-variable, primary-school level questions. The math topics covered include algebra, percentage, fractions, and geometry. There are at most 15 variables and 10 constants. Constants are numbers that do not appear in the question body but the equation in at least five samples. These 10 constants are categorized as follows:

- Geometry: 3.14 and 0.5
- Time: 12, 7, 60
- 0-4: numbers used in geometry, counting, and facts.

Almost all recent deep learning-based MWP solvers are based on the seq2seq implementation, which maps a problem into a *template* first, rather than mapping the problems into equations directly. A *template* is composed of operators and numbers. If the number appears in the problem description, it is a *variable* and is converted into a placeholder according to its position in the description. Otherwise, it is a *constant* and is kept as a number in the template. For example, the template of the problem shown in *Table 1* is $N2-N0-N1$. We get the templates once placeholders replace the numbers in the questions and their correspondence in equations. In ArMATH, there are 883 templates in total. *Table 3* shows the most frequent templates.

| Template | Frequency |
|---|---|
| N0 / N1 | 631 |
| N0 - N1 | 491 |
| N0 * N1 | 481 |
| N1 * N0 | 361 |
| N0 + N1 | 254 |
| N1 / N0 | 245 |
| (N0 * N1) - N2 | 175 |
| N1 + N0 | 162 |
| (N0 / N1) - N2 | 123 |
| (N0 - N1) + N2 | 80 |

Table 3: Top frequent templates in ArMATH

| Dataset | MATH23K | MAWPS | **ArMATH** |
|---|---|---|---|
| Language | Chinese | English | **Arabic** |
| # Questions | 23,160 | 3,320 | **6,000** |
| # Templates | 2,187 | 311 | **883** |
| # Sentences | 70.1K | 6.3K | **11.2K** |
| # Words | 822K | - | **8.5K** |

Table 4: MWP datasets comparison

### 4.1. Datasets Comparison

As discussed earlier, the closest datasets to this work are MATH23K and MAWPS. *Table 4* compares these datasets with the proposed dataset. The information about the other two datasets is taken from their respected papers (Koncel-Kedziorski et al., 2016; Wang et al., 2017b; Wang et al., 2019a). Although the number of words is 8.5k for ArMATH and 822k for MATH23K, the number of unique words used in training after preprocessing is only 2,491 and 3,672 for ArMATH and MATH23K, respectively. In addition, the sentence count in the ArMATH dataset is not very well defined, as Arabic sentences do not end in a period all the time.

*Figure 1* is the histogram comparison of templates for ArMATH and MATH23K. We can see that the top 10 templates account for half the number of samples in both datasets. In addition, the majority of templates appear less than 10 times. This long-tail distribution implies that training MWP solvers for problems with infrequent templates will be much more difficult than those with frequent templates due to the data vacancy. The experimental results reflect this difficulty as well.

## 5. Building Arabic MWP Solvers

### 5.1. Task Definition

One instance of an MWP can be formally presented as a pair $(P, T)$, where $P$ is the problem text and $T$ is the solution expression tree. Specifically, - $P$ is a sequence of word tokens and variables after replacing each number with a placeholder (e.g.,

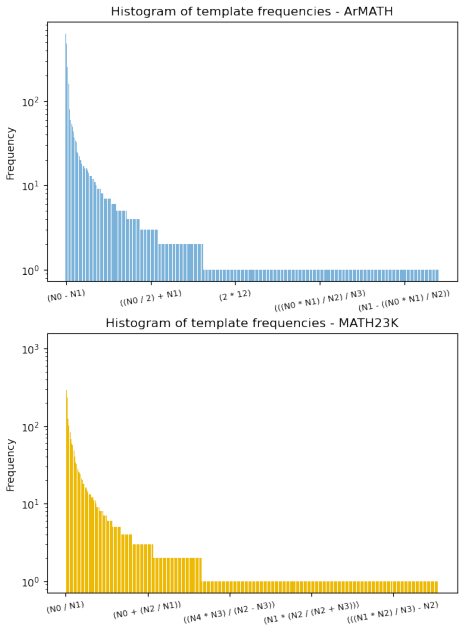


Figure 1: Histogram of the template frequencies for MATH23K and ArMATH datasets

**NUM** for Chinese and English and مجهول for Arabic); because the exact numerical values will be only required later after encoding. The input vocabulary then is simply the collection of word tokens in addition to the placeholder of choice.
- $T$ is the expression tree where numbers that appeared in the question are replaced with an *ordered* placeholder (i.e. $N_x$ denoting the $x^{th}$ variable). Expression trees are an excellent representation for mathematical equations because they ensure the generated sequence's integrity and syntactic correctness by design. The leaf nodes represent numerical values, and inner nodes the operations connecting these numbers. The target language $L^{output}$ is simply the constants found in the questions, in addition to the operations and ordered placeholders, i.e., $L^{output} = L^{const} \cup L^{op} \cup L^{pos}$, where $L^{const}$ is the set of constants tokens, $L^{op}$ is the set of operations, and $L^{pos}$ is the set of positional variables.

Although the target language for all datasets will be the union of the above three, i.e., constants, operations, and positional variables. The exact language will differ depending on the number of constants, operations, and ordered placeholders. For instance, ArMATH contains 10 constants, but MATH23K only has two. *Table 5* shows the output vocabulary for ArMATH and MATH23K. The language is shared except for the extra constants in the ArMATH dataset, highlighted in boldface. The first row shows the operations, the second and third show the constants, and the last rows are the positional variables.

| | | | | |
|---|---|---|---|---|
| $-$ | $/$ | $*$ | $+$ | $\hat{}$ |
| 1 | **2** | 3 | **4** | 3.14 |
| **7** | **0.5** | **60** | **12** | **0** |
| N0 | N1 | N2 | N3 | N4 |
| N5 | N6 | N7 | N8 | N9 |
| N10 | N11 | N12 | N13 | N14 |

Table 5: Output vocabulary for MATH23K and ArMATH. Extra constants in the ArMATH dataset are highlighted in boldface

## 5.2. Arabic MWP Solvers

**The basic GTS solver.** GTS (Xie and Sun, 2019b) is a seq2tree goal-driven model that was initially introduced to solve Chinese MWPs. It has superior performance over other baselines. Therefore, we develop an Arabic MWP solver based on GTS with the training data of ArMATH.

**GTS with pre-trained Arabic word embedding.** The ArMATH dataset size is limited. To promote the problem understanding and handle out-of-vocabulary words in testing samples, we use the pre-trained Arabic word embedding to facilitate the problem encoding. Fasttext (Joulin et al., 2016) and aravec (Soliman et al., 2017) are two pre-trained models for Arabic Word2Vec from Wikipedia data. Although these models are available in a corpse trained over Twitter data too, using wikipedia-trained model was preferred; because Arabic Wikipedia is available in MSA, while tweets are normally written in spoken dialects that differs greatly depending on the author's country. We will employ them in the GTS encoder.

**GTS with transfer learning.** Transfer learning can improve the performance of a target task with fewer data (Torrey and Shavlik, 2009) by leveraging the knowledge to solve tasks in a data-rich source domain. In NLP, it is often used for improving tasks in low-resource languages (target domain) by transferring the knowledge learned from the high-resource input language (source domain) (Zoph et al., 2016). In our setting, the source task is Chinese MWP with 23K samples, and the target task is Arabic MWP with 6K samples in ArMATH. The desired output of the source and target task is similar, as they both output math equations; the difference will generally be the number of variables and the constants. We build the solver by first training GTS over the Chinese MATH23K dataset. Then, the weights will be used to initialize the Arabic solver. There are two possible settings here: 1) transferring the weights for the decoder initialization only; 2) transferring the weights for both encoder and decoder initialization. Then the solver is fine-tuned by the ArMATH dataset.

We will compare the performance of all the solvers in the next section.

# 6. Evaluation & Results

## 6.1. Implementation Setup

For training settings, the default GTS configurations were used initially (Xie and Sun, 2019b): the dimensionality of all hidden states is set to 512 and the dropout rate to 0.5. For the beam search, the beam size is set to 5. The model is trained for 80 epochs with mini-batch size 64 using Adam optimizer (Kingma and Ba, 2017) and cross-entropy loss. The initial learning rate is 0.001 and is divided by two every 20 epochs. The weight decay is set to 0.00001.

## 6.2. Evaluation Metrics

For evaluation, multiple settings were compared against each other, namely, different embedding sizes, different trim counts, and different embedding lookup tables. The accuracy for the equation and answers were recorded after 5-fold cross-validation.

The equation accuracy is the accuracy of generating the same equation as the ground-truth. However, an MWP can usually have multiple correct equations. For example, the following equations are all equivalent.

x * (y + z), x * (z + y), (y + z) * x, (z + y) * x,
x * y + x * z, x * z + x * y, y * x + x * z,
x * z + y * x, y * x + z * x, z * x + y * x

Therefore, answer accuracy is also computed.

## 6.3. Experimental Results

### 6.3.1. The Performance of Arabic MWP solver based on GTS

In the GTS model (Xie and Sun, 2019b), each input word is embedded as a $d$-dim vector (e.g., $d = 128$), and input words that appear less than $t$ times can be converted into an *unknown* word token to improve performance by reducing the number of rare words. In this case, $t$ is referred to as the trim count. We evaluate the performance of the Arabic MWP solver based on GTS trained on the ArMATH dataset when different embedding dimensions $d$ and trimming count $t$ are used. The results are presented in *Table 6*.

The above standard GTS model cannot handle new words and rare words well since it converts them to *unknown*. We employ the pre-trained model aravec (Soliman et al., 2017) and fasttext (Joulin et al., 2016) to replace the word embedding module in GTS. GTS is expected to work well on taking the pre-trained word embedding vectors and optimizing the MWP solver to handle problems with new and rare words. Different versions of embedding in aravec are evaluated, e.g., the usage of n-gram or unigram, the usage of CBOW (continuous bag of words) or SG (skip-gram), and the embedding dimension 100 or 300. The 5-fold results are shown with their confidence intervals. The best model from each group is highlighted in bold text.

The pretrained model aravec (Soliman et al., 2017) and fasttext (Joulin et al., 2016) cannot embed operations. Thus, operations were mapped into their actual names. For instance, the multiplication sign (×) was changed into ضرب (i.e. multiplication).

From *Table 6* we can have the following observations. The performance when no words are trimmed (t=1) for embedding size of 100 is 56.37% for equation accuracy and 68.5% for answer accuracy. Similarly, the equation and answer accuracies are 58.78% and 71.48%, respectively, for embedding dimensionality of 128 and no trimming (t=1). On the other hand, trimming words that appeared less than 7 times improved the performance when the embedding size was 300, with 59% equation accuracy and 71.23% answer accuracy. The best answer accuracy was for embedding dimensionality of 128 and no trimming. However, the best equation accuracy was when the trim count is 7 and embedding dimensionality is 300.

For the aravec models, no clear relationship between different patterns is observed. For instance, an embedding size of 300 is usually better than 100, but that is not the case for the n-gram continuous bag-of-words (CBOW). Similarly, CBOW is better than skip-gram (SG) models, except for the n-gram sg model with embedding size 300. Finally, unigram models are better than their n-gram model only half of the time.

The best aravec model was the CBOW n-gram model with embedding side equals 100; from here on, this model will be referred to as the aravec model for simplicity. The equation and answer accuracies were 58.75% and 71.17%, respectively. In comparison, fasttext model accuracies were 56.30% and 68.58% for equation and answer, respectively.

### 6.3.2. Arabic MAP Solver by Transfer Learning

The motivation here is to transfer the GTS model trained by vast Chinese MWPs to build the GTS model of Arabic MWPs. First, GTS was trained over the Chinese MATH23K dataset, precisely as described in (Xie and Sun, 2019b). Then, the GTS weights were used to initialize the Arabic GTS models. Two settings were tested: transferring the weights of the **decoder only** and transferring the weights of **both the encoder and decoder**. Although the latter can work for closely related languages (Nguyen and Chiang, 2017), Ara-

| Model | | Equation Accuracy | Answer Accuracy |
|---|---|---|---|
| GTS | **$d$=100, $t$=1** | **56.37 ± 2.03** | **68.50 ± 2.00** |
| | $d$=100, $t$=3 | 55.62 ± 0.53 | 67.57 ± 0.90 |
| | $d$=100, $t$=5 | 55.87 ± 2.05 | 67.83 ± 2.33 |
| | $d$=100, $t$=7 | 56.03 ± 1.62 | 68.37 ± 0.87 |
| | **$d$=128, $t$=1** | **58.78 ± 1.30** | **71.48 ± 1.73** |
| | $d$=128, $t$=3 | 58.67 ± 1.33 | 71.42 ± 1.67 |
| | $d$=128, $t$=5 | 58.72 ± 2.20 | 71.08 ± 2.08 |
| | $d$=128, $t$=7 | 58.42 ± 1.25 | 71.20 ± 1.30 |
| | $d$=300, $t$=1 | 57.92 ± 2.67 | 70.30 ± 2.12 |
| | $d$=300, $t$=3 | 57.60 ± 2.98 | 69.73 ± 2.43 |
| | $d$=300, $t$=5 | 57.38 ± 1.53 | 69.77 ± 1.48 |
| | **$d$=300, $t$=7** | **59.00 ± 1.50** | **71.23 ± 0.85** |
| GTS with aravec | **n-grams cbow, $d$=100** | **58.75 ± 2.67** | **71.17 ± 2.58** |
| | n-grams cbow, $d$=300 | 56.80 ± 4.28 | 69.43 ± 4.32 |
| | n-grams sg, $d$=, $d$=100 | 55.97 ± 2.95 | 67.95 ± 3.80 |
| | n-grams sg, $d$=300 | 57.18 ± 1.85 | 70.30 ± 1.13 |
| | unigram cbow, $d$=100 | 56.80 ± 2.70 | 69.23 ± 3.60 |
| | unigram cbow, $d$=300 | 57.12 ± 3.63 | 69.77 ± 3.40 |
| | unigram sg, $d$=100 | 55.70 ± 3.22 | 67.90 ± 4.27 |
| | unigram sg, $d$=300 | 56.22 ± 1.63 | 68.68 ± 2.10 |
| GTS with **fasttext** | | 56.30 ± 1.28 | 68.58 ± 2.25 |

Table 6: Performance of Arabic MWP solver based on GTS trained on ArMATH dataset (without transfer learning). GTS is evaluated with different word embedding dimensionality $d$ and trim size $t$. GTS with Arabic word embedding from **aravec** and **fasttext** are also evaluated. Different versions of embedding in aravec are employed, e.g., the usage of n-gram or unigram, the usage of CBOW (continuous bag of words) or SG (skip-gram), and the embedding dimension 100 or 300. The best model from each group is highlighted in bold text.

| Model | | Equation accuracy | Answer accuracy |
|---|---|---|---|
| GTS aravec | no transfer | 58.75 ± 2.67 | 71.17 ± 2.58 |
| | T-(encoder,decoder) | 59.33 ± 3.57 | 71.97 ± 2.78 |
| | **T-decoder only** | **61.02 ± 1.88** | **74.15 ± 1.77** |
| GTS fasttext | no transfer | 56.30 ± 1.28 | 68.58 ± 2.25 |
| | T-(encoder,decoder) | 56.58 ± 2.25 | 69.35 ± 1.15 |
| | T-decoder only | 59.25 ± 2.10 | 71.95 ± 1.40 |
| GTS embedding d=128 | no transfer | 58.78 ± 1.30 | 71.48 ± 1.73 |
| | T-(encoder,decoder) | 58.18 ± 1.65 | 71.10 ± 0.92 |
| | T-decoder only | 59.88 ± 1.12 | 72.55 ± 1.28 |
| GTS embedding d=300 | no transfer | 59.00 ± 1.50 | 71.23 ± 0.85 |
| | T-(encoder,decoder) | 58.98 ± 1.52 | 71.57 ± 1.42 |
| | T-decoder only | 60.08 ± 1.17 | 72.40 ± 1.27 |

Table 7: Evaluating MWP solvers with and without transfer learning. The decoder weight transfer is better than no-transfer and (encoder,decoder) weight transfer.

bic and Chinese are too different, the performance of transferring the weights of the encoder made the performance suffer.

The overall comparison is presented in *Table 7*. Across all models, the GTS with aravec pretraining and decoder transfer had the best performance, reaching 61.02% and 74.15% for equation and answer accuracies. Comparing the results with and without transfer, the decoder weight transfer does help. Since Arabic and Chinese are very different, it is reasonable that decoder weight transfer is better than both encoder and decoder weight transfer.

*Figure 2* compares decoder-transferred and non-transferred GTS with aravec pretrained embedding on MWPs at different template frequency. The $x$-axis represents the template frequencies in increasing order, while the $y$-axis represents answer accuracy. Each point is a template; textual annotations are provided as examples. The figure shows that transfer learning improved the performance,

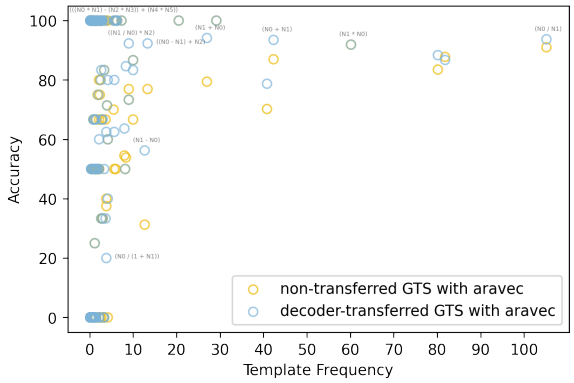


Figure 2: Comparison of decoder-transferred and non-transferred GTS with aravec pretrained embedding on MWPs at different template frequency.

especially for MWPs with templates in a low or medium frequency. This can be well justified because the transferred decoder is more helpful when solving MWPs with fewer samples.

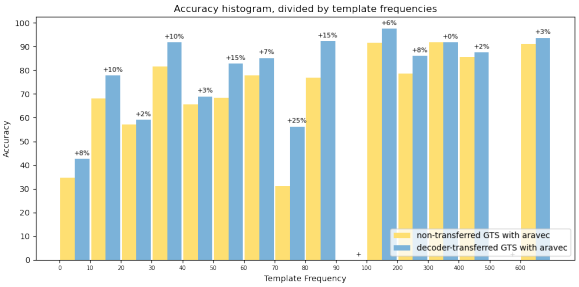


Figure 3: Accuracy compared w.r.t. the template frequency. Decoder-transferred GTS is better than non-transferred GTS, especially on templates with low and median frequency.

To further understand the effectiveness of transfer learning on MWPs compared to the different frequency-level of their templates, we show in *Figure 3* the averaged answer accuracy in varying intervals of template frequencies. For low or medium frequency templates, we can see that transfer learning increased the performance by as much as 25%!
A detailed error analysis is available in the appendix.

## 7. Conclusion and Future Work

In this paper, the first large-scale Arabic Math Word Problem dataset (ArMATH) was collected. It contains 6,000 samples representing 883 templates. In addition, a transfer learning model from Chinese to Arabic was implemented. This work is the first to use deep learning methods to solve Arabic MWP and the first to use transfer learning to promote low-resource MWPs. The accuracy of the model based on transfer learning is 74.15%, which is 3% higher than the baseline that does not use transfer learning. In addition, the accuracy is more than 7% higher than the baseline for templates with few samples representing them. Furthermore, the model can generate new sequences that were not seen before during the training with an accuracy of 27%, 11% higher than the baseline.

For the dataset, more samples can be gathered, translated, or augmented. In terms of the model itself, a model focusing on solving the issues of GTS might be helpful. Namely, the issue of low accuracy in few-shot samples. In addition, models that solve issues in general seq2seq/seq2tree models might work better. Such as using transformer2tree models (Harer et al., 2019) or applying efficient deep learning methods such as compositional learning for translation (Li et al., 2019b).

## 8. Acknowledgements

The research reported in this publication was supported by funding from King Abdullah University of Science and Technology (KAUST).

## References

Amini, A., Gabriel, S., Lin, S., Koncel-Kedziorski, R., Choi, Y., and Hajishirzi, H. (2019a). Mathqa: Towards interpretable math word problem solving with operation-based formalisms. In *NAACL*, pages 2357–2367.

Amini, A., Gabriel, S., Lin, S., Koncel-Kedziorski, R., Choi, Y., and Hajishirzi, H. (2019b). 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, June. Association for Computational Linguistics.

Chiang, T. and Chen, Y. (2018). Semantically-aligned equation generation for solving and reasoning math word problems. In *NAACL*.

Harer, J., Reale, C., and Chin, P. (2019). Tree-transformer: A transformer-based method for correction of tree-structured data.

Hosseini, M. J., Hajishirzi, H., Etzioni, O., and Kushman, N. (2014). Learning to solve arithmetic word problems with verb categorization. In *EMNLP*, pages 523–533.

Huang, D., Shi, S., Lin, C.-Y., and Yin, J. (2017). Learning fine-grained expressions to solve math word problems. In *EMNLP*, pages 805–814.

Joulin, A., Grave, E., Bojanowski, P., Douze, M., Jégou, H., and Mikolov, T. (2016). Fasttext.zip: Compressing text classification models. *CoRR*, abs/1612.03651.

Kingma, D. P. and Ba, J. (2017). Adam: A method for stochastic optimization.

Koncel-Kedziorski, R., Roy, S., Amini, A., Kushman, N., and Hajishirzi, H. (2016). Mawps: A math word problem repository. In *NAACL*, June.

Li, J., Wang, L., Zhang, J., Wang, Y., Dai, B. T., and Zhang, D. (2019a). Modeling intra-relation in math word problems with different functional multi-head attentions. In *ACL*, pages 6162–6167.

Li, Y., Zhao, L., Wang, J., and Hestness, J. (2019b). Compositional generalization for primitive substitutions.

Liang, Z. and Zhang, X. (2021). Solving math word problems with teacher supervision. In *IJCAI*, pages 3522–3528.

Liang, C.-C., Wong, Y.-S., Lin, Y.-C., and Su, K.-Y. (2018). A meaning-based statistical english math word problem solver. In *NAACL*, pages 652–662.

Liu, Q., Guan, W., Li, S., and Kawahara, D. (2019). Tree-structured decoding for solving math word problems. In *EMNLP*, pages 2370–2379.

Miao, S.-Y., Liang, C.-C., and Su, K.-Y. (2020). A diverse corpus for evaluating and developing english math word problem solvers. In *ACL*, pages 975–984.

Mitra, A. and Baral, C. (2016). Learning to use formulas to solve simple arithmetic problems. In *ACL*, pages 2144–2153.

Nguyen, T. Q. and Chiang, D. (2017). Transfer learning across low-resource, related languages for neural machine translation.

Patel, A., Bhattamishra, S., and Goyal, N. (2021). Are nlp models really able to solve simple math word problems?

Qin, J., Lin, L., Liang, X., Zhang, R., and Lin, L. (2020). Semantically-aligned universal tree-structured solver for math word problems. In *EMNLP*.

Saxton, D., Grefenstette, E., Hill, F., and Kohli, P. (2019). Analysing mathematical reasoning abilities of neural models.

Shen, Y. and Jin, C. (2020). Solving math word problems with multi-encoders and multi-decoders. In *COLING*, pages 2924–2934.

Shi, S., Wang, Y., Lin, C.-Y., Liu, X., and Rui, Y. (2015). Automatically solving number word problems by semantic parsing and reasoning. In *EMNLP*, pages 1132–1142.

Siyam, B., Saa, A. A., Alqaryouti, O., and Shaalan, K. (2017). Arabic arithmetic word problems solver. *Procedia Computer Science*, 117:153–160. Arabic Computational Linguistics.

Soliman, A. B., Eissa, K., and El-Beltagy, S. R. (2017). Aravec: A set of arabic word embedding models for use in arabic nlp. *Procedia Computer Science*, 117:256–265. Arabic Computational Linguistics.

Torrey, L. and Shavlik, J. (2009). *Transfer Learning*. University of Wisconsin, Madison WI, USA.

Wang, Y., Liu, X., and Shi, S. (2017a). Deep neural solver for math word problems. In *EMNLP*, pages 845–854.

Wang, Y., Liu, X., and Shi, S. (2017b). Deep neural solver for math word problems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 845–854, Copenhagen, Denmark, September. Association for Computational Linguistics.

Wang, L., Wang, Y., Cai, D., Zhang, D., and Liu, X. (2018). Translating a math word problem to a expression tree. In *EMNLP*, pages 1064–1069.

Wang, L., Zhang, D., Zhang, J., Xu, X., Gao, L., Dai, B. T., and Shen, H. (2019a). Template-based math word problem solvers with recursive neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:7144–7151, 07.

Wang, L., Zhang, D., Zhang, J., Xu, X., Gao, L., Dai, B. T., and Shen, H. T. (2019b). Template-based math word problem solvers with recursive neural networks. In *AAAI*, volume 33, pages 7144–7151.

Wu, Q., Zhang, Q., Fu, J., and Huang, X.-J. (2020). A knowledge-aware sequence-to-tree network for math word problem solving. In *EMNLP*, pages 7137–7146.

Xie, Z. and Sun, S. (2019a). A goal-driven tree-structured neural model for math word problems. In *IJCAI*, pages 5299–5305.

Xie, Z. and Sun, S. (2019b). A goal-driven tree-structured neural model for math word problems. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5299–5305. International Joint Conferences on Artificial Intelligence Organization, 7.

Zhang, J., Lee, R. K.-W., Lim, E.-P., Qin, W., Wang, L., Shao, J., and Sun, Q. (2020a). Teacher-student networks with multiple decoders for solving math word problem. In *IJCAI*, pages 4011–4017.

Zhang, J., Wang, L., Lee, R. K.-W., Bin, Y., Wang, Y., Shao, J., and Lim, E.-P. (2020b). Graph-to-tree learning for solving math word problems. In *ACL*, pages 3928–3937.

Zoph, B., Yuret, D., May, J., and Knight, K. (2016). Transfer learning for low-resource neural machine translation.

Zou, Y. and Lu, W. (2019). Text2math: End-to-end parsing text into math expressions. In *EMNLP*, pages 5330–5340.

# Appendix

## A. Qualitative Results & Error Analysis

The following section compares the qualitative results of the aravec model with no transfer learning against the decoder-only transfer learning model. In terms of the qualitative results, there are certain interesting cases. *Table 8* shows some correct samples randomly selected. The first type of question was an algebraic one. Although both models did not predict the exact equation, their prediction was correct. Then, the next two questions are geometry, one of them is a direct question, and the other is indirect. In addition, both of them use constants. The transfer learning model predicted them both correctly, while the baseline model failed the circle question. The last two questions are general math word problems. The last one is fascinating: the model understood that "all birds flying" = "none is left on the tree".

*Table 9* shows some incorrect samples randomly selected. There are 4 cases of incorrectness observed: first, flipping the operands while using division or subtraction. Unlike multiplication and addition, division and subtraction are not commutative. So $N0 - N1 \neq N1 - N0$. The second error that could occur is over-complication, as shown in the third example. Over-simplification does not seem to be an issue. However, incorrectness due to missing one more constant or operation can happen. Lastly, it can be incorrect because it is simply incorrect.

Finally, *Table 10* shows some interestingly incorrect samples. The first striking error is tricky or ambiguous questions. Depending on the question's meaning, the predicted equation may be correct in the first three examples. Secondly, some errors could occur due to out-of-vocabulary words in the question that convey important concepts. The last row is an example of such a case.

Table 8: Qualitative results - correct samples

| | |
|---|---|
| **Question** | حاصل مجهول في مجهول زائد مجهول .<br>product of NUM * NUM + NUM |
| **Equation** | $((N0 * N1) + N2)$ |
| **aravec no transfer** | $((N1 * N0) + N2)$ \| Correct |
| **aravec T-decoder only** | $((N1 * N0) + N2)$ \| Correct |
| **Question** | اوجد مساحه الدائره اذا علمت ان محيطها هو مجهول سم<br>Find the area of the circle if its circumference is NUM cm |
| **Equation** | $(3.14 * ((N0/(2 * 3.14))\hat{}2))$ |
| **aravec no transfer** | $(3.14 * ((N0/2)\hat{}2))$ \| Incorrect |
| **aravec T-decoder only** | $(3.14 * ((N0/(2 * 3.14))\hat{}2))$ \| Correct |
| **Question** | قطعه ارض مربعه الشكل طول ضلعها مجهول امتار اراد صاحبها ان يبني سورا حولها فكم يبلغ طول هذا السور ؟<br>The length of one side in a square land is NUM m. If the landowner wanted to build a fence around it, how long should it be? |
| **Equation** | $(N0 * 4)$ |
| **aravec no transfer** | $(4 * N0)$ \| Correct |
| **aravec T-decoder only** | $(4 * N0)$ \| Correct |
| **Question** | تدرب عدنان علي ملعب كره القدم مجهول دقيقه في اليوم لمده مجهول ايام في الاسبوع علي مدار مجهول اسابيع فما المده التي قضاها عدنان بالتدريب بالدقائق ؟<br>Adnan practiced football for NUM minutes a day for NUM days in a week for NUM weeks. How many minutes did he spend practicing? |
| **Equation** | $(N2 * (N1 * N0))$ |
| **aravec no transfer** | $((N0 * N1) + N2)$ \| Incorrect |
| **aravec T-decoder only** | $((N0 * N1) * N2)$ \| Correct |
| **Question** | تقف مجهول طيور علي غصن شجره طارت كلها عن الغصن فكم طائرا بقي علي الشجره ؟<br>NUM birds are standing on a tree branch, all of them flew away. How many birds are left on the branch? |
| **Equation** | $(N0 - N0)$ |
| **aravec no transfer** | $(N0 - N0)$ \| Correct |
| **Transfer Learning** | $(N0 - N0)$ \| Correct |

Table 9: Qualitative results - incorrect samples

| | |
|---|---|
| **Question** | يخبز علي مجهول فطيره في مجهول ساعه فكم ساعه يستغرق خبز كل فطيره ؟<br>Ali bakes NUM pies in NUM hours. How long does it take to bake a pie? |
| **Equation** | $(N1/N0)$ |
| **aravec no transfer** | $(N0/N1)$ \| Incorrect |
| **Transfer Learning** | $(N0/N1)$ \| Incorrect |
| **Question** | ما هو قطر الدائره اذا علمت ان مساحتها هي مجهول سم مربع ؟<br>What is the diameter if the circle's area is NUM cm squared? |
| **Equation** | $(2 * ((N0/3.14)\hat{}0.5))$ |
| **aravec no transfer** | $((N0/3.14)\hat{}0.5)$ \| Incorrect |
| **Transfer Learning** | $((N0/3.14)\hat{}0.5)$ \| Incorrect |
| **Question** | يشير الكتاب الاحصائي السنوي لوزاره الصحه لعام مجهول ه الي ان عدد الاطباء في منطقه الرياض من الذكور بلغ مجهول طبيبا ومن الاناث مجهول طبيبات فكم يزيد عدد الاطباء الذكور عن الاناث ؟<br>The official ministry of health's statistics for the year NUM show NUM male doctors in Riyadh and NUM female doctors. By how many is the number of male doctors larger than female ones? |
| **Equation** | $(N1 - N2)$ |
| **aravec no transfer** | $(N0 - (N1 + N2))$ \| Incorrect |
| **aravec T-decoder only** | $(N0 - (N1 + N2))$ \| Incorrect |
| **Question** | مجهول قسمه ( مجهول ناقص مجهول ) ضرب مجهول زائد مجهول ناقص مجهول ) قسمه مجهول .<br>NUM / (NUM - NUM) * NUM + (NUM - NUM) / NUM |
| **Equation** | $(N0/(((N1 - N2) * N3) + ((N4 - N5)/N6)))$ |
| **aravec no transfer** | $(((N0/(N1 - N2)) + ((N1 - N2) * N3))/(N5/N6))$ \| Incorrect |
| **aravec T-decoder only** | $(((N0/(N1 - N2)) - (N2 * N3))/N6)$ \| Incorrect |

Table 10: Qualitative results - interesting incorrect samples

| | |
|---|---|
| **Question** | انتقل خالي الي شقه جديده اذا كان هنالك مجهول طوابق فوقه و مجهول طوابق تحته فكم طابقا في هذه العماره ؟<br>My uncle moved to a new apartment. If there are NUM floors above him and NUM floors below him. How many floors are there in this building? |
| **Equation** | $((N0 + N1) + 1)$ |
| **aravec no transfer** | $(N0 + N1)$ \| Incorrect |
| **aravec T-decoder only** | $(N0 + N1)$ \| Incorrect |
| **Question** | مجهول بعد زياده مجهول مرات =<br>NUM after increasing NUM times = |
| **Equation** | $(N0 * (N1 + 1))$ |
| **aravec no transfer** | $(N0 * N1)$ \| Incorrect |
| **aravec T-decoder only** | $(N0 * N1)$ \| Incorrect |
| **Question** | اوجد ناتج : مجهول مطروح منها مجهول مقسومه علي مجهول ؟<br>Find the answer: NUM minus NUM divided by NUM |
| **Equation** | $(N0 - (N1/N2))$ |
| **aravec no transfer** | $((N0 - N1)/N2)$ \| Incorrect |
| **aravec T-decoder only** | $((N0 - N1)/N2)$ \| Incorrect |
| **Question** | لدي مني ورقتان من فئه مجهول ريالات ولدي اخيها خالد و مجهول اوراق من فئه مجهول ريالات . . من فئه الريال مجهول فما مجموع ما لديهما ؟<br>Muna has two NUM $ notes. Her brother Khalid NUM NUM $ notes and UNK UNK of NUM $. What is the total of what they have? |
| **Equation** | $(((2 * N0) + (N1 * N2)) + (2 * N3))$ |
| **aravec no transfer** | $(((2 * 2) + 1) + (N2 * N3))$ \| Incorrect |
| **aravec T-decoder only** | $(((2 + 1) + 1) + (2 * N3))$ \| Incorrect |