SemEval-2021 Task 10: Source-Free Domain Adaptation for Semantic Processing

Egoitz Laparra\textsuperscript{1} \hfill Xin Su\textsuperscript{1} \hfill Yiyun Zhao\textsuperscript{1} \\
Özlem Uzuner\textsuperscript{2} \hfill Timothy A Miller\textsuperscript{3} \hfill Steven Bethard\textsuperscript{1}

\textsuperscript{1}University of Arizona, Tucson, AZ 85721, USA  \\
{laparra, xinsu, yiyunzhao, bethard}@email.arizona.edu  \\
\textsuperscript{2}George Mason University, Fairfax, VA 22030, USA  \\
ozuner@gmu.edu  \\
\textsuperscript{3}Boston Children’s Hospital and Harvard Medical School, Boston, MA 02115, USA  \\
timothy.miller@childrens.harvard.edu

Abstract

This paper presents the Source-Free Domain Adaptation shared task held within SemEval-2021. The aim of the task was to explore adaptation of machine-learning models in the face of data sharing constraints. Specifically, we consider the scenario where annotations exist for a domain but cannot be shared. Instead, participants are provided with models trained on that (source) data. Participants also receive some labeled data from a new (development) domain on which to explore domain adaptation algorithms. Participants are then tested on data representing a new (target) domain. We explored this scenario with two different semantic tasks: negation detection (a text classification task) and time expression recognition (a sequence tagging task).

1 Introduction

Data sharing restrictions are common in NLP datasets. For example, Twitter policies do not allow sharing of tweet text, though tweet IDs may be shared. The situation is even more common in clinical NLP, where patient health information must be protected, and annotations over health text, when released at all, often require the signing of complex data use agreements.

The Source-Free Domain Adaptation shared task presents a new framework that asks participants to develop semantic annotation systems in the face of data sharing constraints. A participant’s goal is to develop an accurate system for a target domain when annotations exist for a related domain but cannot be distributed. Instead of annotated training data, participants are given a model trained on the annotations. Then, given unlabeled target domain data, they are asked to make predictions. This is a challenging setting, and much previous work on domain adaptation does not apply, as it assumes access to source data (Ganin et al., 2016; Ziser and Reichart, 2017; Saito et al., 2017; Ruder and Plank, 2018), or assumes that labeled target domain data is available (Daumé III, 2007; Xia et al., 2013; Kim et al., 2016; Peng and Dredze, 2017).

Two different semantic tasks in English were created to explore this framework: negation detection and time expression recognition. These represent two common types of classification tasks: negation detection is typically formulated as predicting an attribute of a word or span given its context, and time expression recognition is typically formulated as a named entity tagging problem. Both of these tasks have previously been run as shared tasks, and had at least two different domains of data available, and we had access to experienced annotators for both tasks, allowing us to annotate data in a new domain.

Negation detection is the task of identifying negation cues in text. This task has been widely studied by previous work (Chapman et al., 2007, 2001; Harkema et al., 2009; Sohn et al., 2012) including the development of a variety of datasets (Uzuner et al., 2011; Mehrabi et al., 2015). However, there are still large performance losses in the cross-domain setting (Wu et al., 2014).

For negation detection, we provided a “span-in-context” classification model, fine-tuned on instances of the SHARP Seed dataset of Mayo Clinic clinical notes, which the organizers have access to but cannot currently be distributed. (Models were approved to be distributed, as the data is de-identified.) In the SHARP data, clinical events are marked with a boolean polarity indicator, with values of either asserted or negated. As development
data, we used the i2b2 2010 Challenge Dataset, a de-identified dataset of notes from Partners HealthCare. The evaluation dataset for this task consisted of de-identified intensive care unit progress notes from the MIMIC III corpus (Johnson et al., 2016).

Time expression recognition has been a key component of previous temporal language related competitions, like TempEval 2010 (Pustejovsky and Verhagen, 2009) and TempEval 2013 (UzZaman et al., 2013). For this task, we followed the Compositional Annotation of Time Expressions (SCATE) schema (Bethard and Parker, 2016) used in in SemEval 2018 Task 6 (Laparra et al., 2018). As in negation detection, previous works have also observed a significant performance degradation on domain shift (Xu et al., 2019).

For time expression recognition, we provided a sequence tagging model, fine-tuned on de-identified clinical notes from the Mayo Clinic, which were available to the task organizers, but are difficult to gain access to due to the complex data use agreements necessary. (Models were approved to be distributed, as the data is deidentified.)

The development data was the annotated news portion of the SemEval 2018 Task 6 data whose source text is from the freely available TimeBank. For evaluation, we used a set of annotated documents extracted from food security warning systems.

The main impact of this task is to drive the NLP community to address the serious challenges of data sharing constraints by designing new domain adaptation algorithms that allow source data and target data to remain separate, rather than assuming they can be shared freely with each other.

## 2 Data and Resources

In this section, we describe both negation detection and time expression recognition tasks, the models fine-tuned on a difficult-to-obtain set of annotated data, the development data representing a new domain on which participants can explore their approaches for domain adaptation, and the test data representing another new domain on which the systems developed by participants are evaluated. Details of the different data sets can be found in Tables 1 and 2.

### 2.1 Negation detection

The negation detection track asks participants to classify clinical event mentions (e.g., diseases, symptoms, procedures, etc.) for whether they are being negated by their context. This task can be treated as a “span-in-context” classification problem, where the model jointly considers both the event to be classified and its surrounding context. For example, the sentence:

\[(1) \text{Has no diarrhea and no new lumps or masses}\]

has three relevant events (diarrhea, lumps, masses), two cue words (both no), and all three entities are negated. This task is important in the clinical domain because it is common for physicians to document negated information encountered during the clinical course, for example, when ruling out certain elements of a differential diagnosis.

This task can be treated as a “span-in-context” classification problem, where the model jointly considers both the event to be classified and its surrounding context. For example, a typical transformer-based encoding of this problem for the diarrhea event in the example above looks like:
(2) Has no \(<e>\) diarrhea \(</e>\) and no new lumps or masses.

**Pre-trained model** Participants were provided with a “span-in-context” classification model, trained on the 10,259 instances (902 negated) in the SHARP Seed dataset of de-identified clinical notes from Mayo Clinic, which the organizers had access to but cannot currently be distributed. In the SHARP data, clinical events are marked with a boolean polarity indicator, with values of either ASSERTED or NEGATED.

**Development data** Participants could use as development data the i2b2 2010 Challenge Dataset, a de-identified dataset of notes from Partners Healthcare, containing 5,545 entities labeled with an assertion status in the set \{ASSERTED, NEGATED, UNCERTAIN, HYPOTHETICAL, CONDITIONAL, FAMILYRELATED\}. We provided scripts that extracted i2b2 entities and simplified the label set to \{NEGATED, NOTNEGATED\}. Since the i2b2 2010 dataset consisted of notes from two sources, Partners and MIMIC III, the latter of which overlaps with our proposed test set, our script also filtered the development instances to contain only those from the Partners notes.

**Test data** During the testing period, participants were provided with the raw text of 622,703 instances drawn from the MIMIC III corpus\(^1\), which contains manually de-identified progress notes for patients from the intensive care unit of Beth Israel Deaconess Medical Center, with entities of interest already identified. From this, we manually annotated 9,580 instances of which 958 were negated.

### 2.2 Time expression recognition

The time expression recognition track, which represents a sequence-tagging task, uses the fine-grained time expression annotations that were a component of SemEval 2018 Task 6 (Laparra et al., 2018). For example:

\[
\begin{array}{c}
\text{MONT}-\text{H} \text{-} \text{O} \text{-} \text{F} \text{-} \text{Y} \text{E} \text{R} \text{Y} \\
\text{JANUARY} \text{ OF} \text{ 2009} \\
\text{NUMBER} \text{ PERIOD} \text{ AFT} \text{ER} \text{ HOURS} \text{ AFTER} \text{ HER} \text{ MEAL.}
\end{array}
\]

This task can be treated as a sequence classification problem, as in other named-entity tagging tasks.

---

\(^1\)https://mimic.physionet.org/

**Pre-trained model** Participants were provided with a sequence tagging model, trained on the 18,020 time expressions in the clinical portions of the SemEval 2018 Task 6, that were available to the task organizers, but are currently difficult to gain access to due to the complex data use agreements.

**Development data** Participants could use as development data the annotated news portion of the SemEval 2018 Task 6 data. The source text is from the freely available TimeBank\(^2\), and the 2,231 time entity annotations were from the freely available SCATE GitHub repository\(^3\).

**Test data** During the testing period, participants were provided with the raw text of 47 reports drawn from food security warning systems\(^4\) and asked to predict time expressions. From this, we used 17 documents that included 1,900 time entities, annotated by two independent annotators and an adjudicator.

### 3 Evaluation Metrics

Negation detection was evaluated using the precision/recall/F\(_1\) of the negated class, as used in most published work. Time expression recognition was evaluated using the standard precision/recall/F\(_1\) previously used for the entity-finding portion of SemEval 2018 Task 6.

In both cases, the metrics are defined as:

\[
P(S, H) = \frac{|S \cap H|}{|S|} \\
R(S, H) = \frac{|S \cap H|}{|H|} \\
F_1(S, H) = \frac{2 \cdot P(S, H) \cdot R(S, H)}{P(S, H) + R(S, H)}
\]

where \(S\) is the set of items predicted by a system and \(H\) is the set of items manually annotated by humans.

### 4 Baseline Systems

To provide a comparison benchmark, we proposed two baselines for both negation detection and time expression recognition:

\(^2\)https://www.cs.york.ac.uk/semeval-2013/task1/index.php?id=data.html

\(^3\)https://github.com/bethard/anafora-annotations

\(^4\)Like the UN World Food Programme https://www.wfp.org/ or the Famine Early Warning Systems Network https://fews.net/.
Table 3: Performance of the baselines on the source domain, where Source-Trained (Src-Trained) was trained, and the two target domains (dev and test). For Dev-Tuned, dev set was also used for training.

| Sub-task               | System     | source | dev | test |
|------------------------|------------|--------|-----|------|
|                        |            | $P$    | $R$ | $F_1$ |
| Negation               | Src-Trained| 0.820  | 0.851| 0.818| 0.834| 0.917| 0.516| 0.660|
| Negation               | Dev-Tuned  |        | -   | -    | -    | 0.908| 0.611| 0.730|
| Time Expression        | Src-Trained| 0.967  | 0.968| 0.968| 0.775| 0.768| 0.771| 0.849| 0.746| 0.794|
| Time Expression        | Dev-Tuned  |        | -   | -    | -    | 0.827| 0.782| 0.804|

**Source-Trained** Models pre-trained on only the source train data, i.e., the models that the organizers shared with the participants as explained in Section 2.

**Dev-Tuned** Models pre-trained on the source data (i.e., Source-Trained) and then fine-tuned on the labeled dev data.

All baselines were built on RoBERTa (Liu et al., 2019) using the HuggingFace Transformers library.\(^5\)

Table 3 shows the performance of the baselines on negation detection and time expression recognition respectively. In both cases, there is a big drop in the performance of Source-Trained when it is applied to out-of-domain datasets. Using the development data to continue training the model (Dev-Tuned) provides some improvement for both tasks, but it is still far from in-domain performance.

5 Participating Systems

Since our goal was to see a set of experiments as varied as possible, we did not impose any constraint on the approaches participants could submit, including the use of any of the unlabeled or labeled data provided. The task had 9 participants that submitted 20 unique runs in total, as shown in Table 4. For each task, 2 submissions per team were allowed. There were 5 participants and 8 submission in negation detection, and 7 participants and 12 submissions in time expression recognition. Only 3 participants took part in both tasks.

5.1 Negation detection

**BLCUFIGHT-1** tried a self-training method fixing the top classifier so only the feature extractor was updated. Then, they ran an ensemble of 3 models. **BLCUFIGHT-2** built an unlabeled dataset selecting 2,000 instances from the development set, 2,000 from the test set and 2,886 from the training set. They used that unlabeled dataset progressively to continue fine-tuning the distributed model (for 2 epochs) following a self-learning approach. They additionally selected some negative prefixes and negative words as rules. The final predictions were obtained from an ensemble of 5 models.

**UArizona-1** used the development data to continue fine-tuning the distributed model (for 10 epochs). Then, they randomly sampled 3,000 examples from unlabeled test data and performed 2 self-learning iterations, using a 0.95 threshold to filter the pseudo training examples.

**IITK-1** also adapted the model with pseudo labels obtained from a sample of 25,000 instances from the test data. They selected predictions with low entropy as the pseudo training examples, performed data-augmentation on the selected instances, and used the resulting set to continue training the distributed model. **IITK-2** applied an adaptive version of this approach by slowly increasing the entropy threshold after each epoch and filtering again the training instances.

**MedAI-1** and **MedAI-2** followed a self-learning strategy preceded by a negation-aware pre-training process. For the latter, they built a dataset applying some heuristics on the test data. First, they manually collected a dictionary including negation cues, such as “not”, “no”, “no longer”. Second, they selected the nouns within a 3 token window around occurrences of the negation cues. Finally, they labeled the cue-noun pairs as negated instances.

**Observations:** Self-learning was the most widely applied technique (6 out of 8 submissions), 3 submissions extended this with heuristics, 2 submissions extended it with data augmentation, and 2 applied it with a model ensemble. Only 2 submissions leveraged the development set of which only 1 used the labeled data. All the

---

\(^5\) https://github.com/huggingface/transformers.
submissions but one used the unlabeled test data to produce a training set for the target domain, either in the form of pseudo-labeled instances (5 submissions) or by heuristic-driven annotation (2 submissions). No submissions used additional resources.

5.2 Time expression recognition

**BLCUFIGHT-1** and **BLCUFIGHT-2** proposed an unsupervised mean-teacher framework that updates the model in a self-learning manner. Additionally, they used a set of string-matching heuristics derived from the development set, e.g., “spring” or “summer” for Season-Of-Year, and “decades” for Period. **BLCUFIGHT-1** ensembled 2 models for a better robustness.

**Self-Adapter-1** and **Self-Adapter-2** generated pseudo training examples by running the provided model on the test documents and selecting the sentences where the highest words’ entropy was lower than 0.1. In **Self-Adapter-1**, they combined the predictions of both a fixed version and a trainable version of the model. **Self-Adapter-2** used only the trainable model. In both submissions, the trainable model was updated by applying 3 iterations of the *sloughing trick*, i.e., training the model iteratively with the pseudo-labels obtained by the model of the previous iteration.

**PTST-UoM-1**, also following a self-training approach, built, for each unlabeled input sentence, a chart containing high probability label sequences produced by the distributed model and applied it as a supervision signal. They used the labeled development data for tuning some of the hyperparameters.

**UArizona-1** combined active learning and data augmentation. They ran 5 iterations of the following steps: 1) predict the unlabeled test data and then select 32 sentences with high entropy calculated as the sum of the entropy of all tokens in the sentence; 2) manually label time entities in the 32 sentences; 3) for each manually labeled time entity, generate
5 additional training examples using 5 new words with same entity type; 4) train the model on the resulting dataset. The same method was used by UArizona-2, but, in this case, they fixed some errors in the manual annotations.

KISNL-1 and KISNL-2 used the development labeled data as a fine-tuning resource, which was complemented by a data augmentation process. They did not use the unlabeled test data, nor any other resource.

YNU-HPCC-1 and YNU-HPCC-2 also used the labeled portion of the development set. They fine-tuned 4 popular transformer-based pre-trained models: RoBERTa, BERT (Devlin et al., 2019), DistilBERT (Sanh et al., 2020) and ALBERT (Lan et al., 2020). The final prediction was given by hard voting strategy, integrating the results of the 4 models along with Source-Trained.

Observations: Self-learning (5 submissions) and data augmentation (4 submissions) were the most commonly followed approaches. 2 submissions extended a self-learning technique with manually created heuristics. Only 3 submissions proposed ensemble methods. In this task, the development set was more frequently exploited and 4 submissions made use of the labeled data to continue fine-tuning the provided model. The test set was manually annotated by 2 submissions that followed an active learning approach, along with some additional resources. 4 submissions did not use the unlabeled test data.

6 Evaluation Results

Tables 5 and 6 shows the performance of the systems described in Section 5 on negation detection and time expression recognition. For comparison, the tables also include the performance of the baselines described in Section 4.

6.1 Negation detection

As shown in Table 5, 7 out of 8 submissions on negation detection outperform Source-Trained but only 4 performed better than Dev-Tuned.

The best results were obtained by MedAI-1 and MedAI-2, achieving 16.2 and 9.2 percentage points of \( F_1 \) more than Source-Trained and Dev-Tuned, respectively. These model had a large recall improvement (14.5 points more than Source-Trained and 24.0 more than Dev-Tuned) at the expense of a slight degradation in precision.

| System          | \( P \) | \( R \) | \( F_1 \) |
|-----------------|--------|--------|-------|
| MedAI-1\( ^\dagger \) | 0.902  | 0.756  | 0.822 |
| MedAI-2\( ^\dagger \) | 0.902  | 0.756  | 0.822 |
| UArizona-1\( ^\dagger \) | 0.880  | 0.680  | 0.767 |
| BLCUFIGHT-2\( ^\dagger \) | 0.913  | 0.616  | 0.736 |
| IITK-2\( ^\dagger \) | 0.876  | 0.624  | 0.729 |
| Boom-1          | 0.929  | 0.597  | 0.727 |
| IITK-1\( ^\dagger \) | 0.939  | 0.566  | 0.706 |
| BLCUFIGHT-1     | 0.528  | 0.639  | 0.578 |
| Dev-Tuned       | 0.908  | 0.611  | 0.730 |
| Source-Trained  | 0.917  | 0.516  | 0.660 |

Table 5: Official results (ranked by \( F_1 \)) on negation detection. Superscripts indicate that the submission used: \( ^\dagger \)unlabeled dev, \( ^\dagger \)labeled dev or \( ^\dagger \)unlabeled test data

\( \text{HITK-1} \) and \( \text{Boom-1} \) outperform both baselines in terms of precision but obtain a worse recall than Dev-Tuned.

The 3 best submissions on this task (MedAI-1, MedAI-2 and UArizona-1) make use of some kind of labeled data. In the case of MedAI-1 and MedAI-2, this data belongs to the target test domain, which could explain the good results of these 2 submissions. BLCUFIGHT-2, the next best performing system and the only other one that outperforms both baselines, also applies some manual supervision in the form of hand-crafted rules.

In general, self-learning proved to be an effective technique for negation detection, especially in terms of recall, while data-augmentation also shows recall improvements in some cases. As usual, ensemble models are helpful. Including some manual supervision drove the largest gains.

6.2 Time expression recognition

Table 6 shows that for time expression recognition, 9 out of 12 submissions outperformed Source-Trained and only 3 obtained a better performance than Dev-Tuned. The gains were generally smaller than on negation detection, with the best models being only 2.1 percentage points of \( F_1 \) above Source-Trained and 1.1 percentage points above Dev-Tuned.

As in negation detection, the best performing system (BLCUFIGHT-1) utilizes some form of manual supervision. In this case, they apply a set of manually created string matching heuristics in combination with a self-learning approach that is boosted by a model ensemble.

In this task, the use of the labeled development
Table 6: Official results (ranked by $F_1$) on time expression recognition. Superscripts indicate that the submission used: *unlabeled dev, +labeled dev or †unlabeled test data

| System           | P     | R     | $F_1$ |
|------------------|-------|-------|-------|
| BLCUFIIGHT-1†    | 0.847 | 0.785 | 0.815 |
| Self-Adapter-1†  | 0.873 | 0.757 | 0.811 |
| BLCUFIIGHT-2†    | 0.834 | 0.787 | 0.810 |
| YNU-HPCC-2†      | 0.817 | 0.791 | 0.803 |
| Self-Adapter-2†  | 0.839 | 0.760 | 0.797 |
| PTST-UoM-1†      | 0.901 | 0.713 | 0.796 |
| UArizona-1†      | 0.786 | 0.804 | 0.795 |
| UArizona-2†      | 0.783 | 0.807 | 0.795 |
| Boom-1           | 0.869 | 0.732 | 0.795 |
| KISNLP-1†        | 0.810 | 0.777 | 0.793 |
| KISNLP-2†        | 0.798 | 0.764 | 0.781 |
| YNU-HPCC-1†      | 0.872 | 0.655 | 0.748 |
| Dev-Tuned        | 0.827 | 0.782 | 0.804 |
| Source-Trained   | 0.849 | 0.746 | 0.794 |

The set is more frequent. 5 of the submissions made use of this data, but none obtained better results than Dev-Tuned, although YNU-HPCC-2 got a close $F_1$ score. In the case of PTST-UoM-1, this explained by the fact that they only consulted this set to fine-tune the hyperparameters of their model, although this strategy was enough to obtain the best precision among all systems. The approach of KISNLP-1 and KISNLP-2 is the same as Dev-Tuned but combined with some data-augmentation, resulting in a drop in performance. This may be caused by only using the development set to perform the augmentation since, after all, it belongs to a different domain than the test documents. YNU-HPCC-2 is the only submission, along with YNU-HPCC-1, that utilized other pre-trained transformers, in an ensemble mode, besides the model provided.

UArizona-1 and UArizona-2 are the only submissions that tried an active learning strategy. The approach performed slightly better than Source-Trained but worse than Dev-Tuned. This contrasts with the best performing model on negation detection that also implemented a manual annotation process on test data, but it is explained by the much more complex annotation scheme of time expressions. UArizona-2 obtains the best recall on the task.

Self-Adapter-1 is the only submission that outperforms Dev-Tuned without using any kind of manual supervision. The only difference with respect to Self-Adapter-2, that did not perform as well, is that the original model trained on the source domain is consulted to produce pseudo-examples in every iteration of their self-learning technique. This seems to counteract a possible degradation of the predictions caused by updating the model with pseudo-labels.

7 Future directions

Self-learning and data augmentation were the most frequently used techniques. Some systems, including the best performing ones, incorporated some kind of manual supervision in the form of active-learning, hand-crafted heuristics or semi-automatically building a training set. This suggests that future work on source-free domain adaptation will focus on acquiring data instances for the target domain either automatically or manually, and use such data to continue fine-tuning the source-domain model.

Any new approaches will have to address some fundamental challenges. Errors in the generation of pseudo-labels propagate in successive self-learning iterations degrading the performance. Continual fine-tuning on data from a new domain can lead to catastrophic forgetting, especially if the data is restricted to certain instances like those drawn from high-confident predictions of the source model. Manually supervised approaches, such as active learning, do not necessarily solve these problems due to the complexity of some annotation schemes, like in time expressions recognition, and the reduced number of labels that this methods can yield.

Some of the experiments carried out during this task have approached these issues and should be taken as an starting point for future research.

8 Conclusion

In this paper, we have described the Source-Free Domain Adaptation shared task held within SemEval-2021. In this task, participants were asked to adapt a given model to a target domain when the access to both labeled and unlabeled source data is restricted. In contrast to previous tasks on domain adaptation, participants were only provided with a trained model and the target unlabeled data. Systems were evaluated on two tasks, negation detection and time expression recognition, that are paradigmatic examples of two common types of machine-learning problems in natural language processing: text classification and sequence
9 participants took part in the challenge with 20 different systems. In negation detection, 8 submissions were received from 5 participants while 7 participants submitted 12 runs for time expression recognition. 3 participants presented approaches for both tasks. 7 out of 8 submissions for negation detection and 9 out of 12 submissions for time expression recognition outperformed the model trained on the source domain. Compared to the same model fine-tuned on the development data, 4 systems in negation detection and 3 in time expression recognition showed a better performance.

This is the first time that such a framework is formally designed and aims to draw the community’s attention to a challenging problem that seriously affects the deployment of NLP models to real-life scenarios, like health institutions.

The scripts and the code of the baselines, along with the development and test data, can be obtained from the task’s GitHub repository. The trained models are available in the HuggingFace model hub for both negation detection and time expression recognition. The CodaLab leader-board of the of the post-evaluation phase will continue to accept submissions indefinitely.

Acknowledgments

Research reported in this publication was supported by the National Library of Medicine of the National Institutes of Health under Award Numbers R01LM012918 and R01LM010090. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

References

Steven Bethard and Jonathan Parker. 2016. A semantically compositional annotation scheme for time normalization. In Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC’16), pages 3779–3786, Portorož, Slovenia. European Language Resources Association (ELRA).

Wendy Chapman, John Dowling, and David Chu. 2007. ConText: An algorithm for identifying contextual features from clinical text. In Biological, translational, and clinical language processing, pages 81–88, Prague, Czech Republic. Association for Computational Linguistics.

Wendy W. Chapman, Will Bridewell, Paul Hanbury, Gregory F. Cooper, and Bruce G. Buchanan. 2001. A simple algorithm for identifying negated findings and diseases in discharge summaries. Journal of Biomedical Informatics, 34(5):301–310.

Hal Daumé III. 2007. Frustratingly easy domain adaptation. In Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, pages 256–263, Prague, Czech Republic. Association for Computational Linguistics.

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.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. Journal of Machine Learning Research, 17(59):1–35.

Henk Harkema, John N. Dowling, Tyler Thorblade, and Wendy W. Chapman. 2009. ConText: An algorithm for determining negation, experiencer, and temporal status from clinical reports. Journal of Biomedical Informatics, 42(5):839–851. Biomedical Natural Language Processing.

Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-Wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. 2016. MIMIC-III, a freely accessible critical care database. Scientific data, 3(1):1–9.

Young-Bum Kim, Karl Stratos, and Ruhi Sarikaya. 2016. Frustratingly easy neural domain adaptation. In Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pages 387–396, Osaka, Japan. The COLING 2016 Organizing Committee.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In International Conference on Learning Representations.

Egoitz Laparra, Dongfang Xu, Ahmed Elsayed, Steven Bethard, and Martha Palmer. 2018. SemEval 2018 task 6: Parsing time normalizations. In Proceedings
