A Deep Learning and Knowledge Transfer Based Architecture for Social Media User Characteristic Determination

Matthew Riemer, Sophia Krasikov, and Harini Srinivasan
IBM T.J. Watson Research Center
1101 Kitchawan Road
Yorktown Heights, NY 10598, USA
{mdriemer, kras, harini}@us.ibm.com

Abstract

Determining explicit user characteristics based on interactions on Social Media is a crucial task in developing recommendation and social polling solutions. For this purpose, rule based and N-gram based techniques have been proposed to develop user profiles, but they are only fit for detecting user attributes that can be classified by a relatively simple logic or rely on the presence of a large amount of training data. In this paper, we propose a general purpose, end-to-end architecture for text analytics, and demonstrate its effectiveness for analytics based on tweets with a relatively small training set. By performing unsupervised feature learning and deep learning over labeled and unlabeled tweets, we are able to learn in a more generalizable way than N-gram techniques. Our proposed hidden layer sharing approach makes it possible to efficiently transfer knowledge between related NLP tasks. This approach is extensible, and can learn even more from metadata available about Social Media users. For the task of user age prediction over a relatively small corpus, we demonstrate 38.3% error reduction over single task baselines, a total of 44.7% error reduction with the incorporation of two related tasks, and achieve 90.1% accuracy when useful metadata is present.

1 Introduction

Two major Social Media Analytics use cases that are driving business value for businesses today are social recommendation systems and social polling applications.

Social recommendation systems analyze attributes of Social Media users and historical trends to recommend personalized products and advertisements to users. The accuracy and robustness of these systems has a direct impact on user satisfaction and ROI, making improvement of these systems a very worthwhile area of study.

Social polling refers to effectively carrying out massive surveys over Social Media. Organizations find applications with these capabilities useful for brand management, campaign management, and understanding key social trends. State of the art social polling systems include a capability of measuring trending topics and sentiment. These systems also include a capability to analyze the user characteristic level dependencies of these trends. For this use case, informative characteristics for businesses to analyze may include a user's age range, gender, ethnicity, income range, location, hobbies, political leanings, and brand affinities. Additionally, both high precision and high recall for all features is paramount to the success of these systems. Low precision or low recall for user attributes skew trends seen over aggregate data, and defeat the purpose of using these solutions to discover statistically founded business insights.

Social Media organizations, generally with strong inherent privacy restrictions, like Facebook have access to many user level characteristics that have been directly inputted to the website. However, there is great interest in analyzing these same kinds of qualities on more public platforms like Twitter and Blogs, where comments are more rea-

Proceedings of SocialNLP 2015@NAACL-HLT, pages 39–47,
Denver, Colorado, June 5, 2015.

dily accessible to organizations interested in Social Media analytics. In this situation, text analytics techniques are commonly used to infer qualities about these users that are not explicitly provided to organizations analyzing this content.

The difficulty of extracting a characteristic about a user based on tweets alone varies greatly by the type of characteristic. NLP rule based approaches (Krishnamurthy et al., 2009) have been commonly used as a means to perform micro-segment analysis of Social Media users. These techniques have been very effective at creating extractors for user attributes like "fan of" relationships, and gender determination with the presence of very little training data. For example, by knowing the key characters, actors, and plot details of a TV show, the logic is intuitive for making an individual rule based extractor that monitors expressed interest by a Social Media user in that show. Moreover, a gender prediction system can be made pretty reliable simply by extracting profile first names, and matching to large lists of female and male names. However, rule based techniques are not good solutions for analyzing more subtle relationships in social posts like those needed for predicting a user's age range, income range, or political leanings. Additionally, as social trends change and users age, it is very desirable for classifiers focused on these tasks to be adaptive and have the ability to efficiently relearn from scratch.

As such, Machine Learning techniques make sense as a means for creating classifiers of more complex user characteristics. N-gram based techniques have commonly been applied to social media analytics problems (Go et al., 2009), (Kökciyan et al., 2013), and (Speriosu et al., 2011). However, we have found that these techniques are not effective without a substantial amount of supervised training data or an extremely reliable semi-supervised method of creating a stand-in corpus.

In this paper, we propose an end-to-end architecture to address the key problems exhibited with common NLP techniques in analyzing subtly expressed social media user characteristics. We will demonstrate our architecture's effectiveness at predicting user age based on a modest 1266 user training set compiled by a team of four researchers in a few hours of work for each person manually annotating data. Our end-to-end method improves on N-gram machine learning techniques by:

1. Building unsupervised text representations that naturally pick up semantic and syntactic synonymy relationships.
2. Effectively utilizing knowledge acquired from unlabelled data.
3. Taking advantage of powerful deep neural networks to increase prediction accuracy.
4. Leveraging a practical framework for transferring knowledge between related user characteristic classifiers for increased performance without increasing the number of free parameters.
5. Establishing a methodology for efficient knowledge transfer from structured metadata related to a user.

Although our main intent is to show the effectiveness of our architecture for Social Media analytics use cases, there is little about our system that has virtues specific to the social media domain. Considering the collection of a user's historical tweets as equivalent to a text document, our approach can serve as a general purpose text analytics architecture, especially for use cases with limited training data. In fact, tweets are generally regarded as more challenging to analyze than other text because of the noisy language and ambiguous content.

The rest of the paper is organized as follows: In Section 2, we describe our data set and go over our experimental methodology. Section 3 gives an overview of the benefits we see by exploring unsupervised text vector techniques. In Section 4 we explain the benefit of building deep learning models on top of unsupervised features. We proceed to explain popular multitask deep learning techniques and their failures for our problem statement in Section 5. Section 6 is an overview of our hidden layer sharing approach, which we validate in Section 7. Section 8 explains how our model is extensible for the incorporation of structured metadata. Finally, Section 9 concludes the paper.

2 Experimental Methodology

Without access to any reliable user provided age information, we had to rely on human judgment to create gold standard annotations for the ages of users on Twitter. We randomly generated Twitter usernames and had a team of four people manually go to Twitter.com and look at their profile. The instructions were to look at the user's Twitter pro-

file including pictures and their tweets to judge their age range and discard any users for whom the age range was not clear. The annotators looked through the user's recent tweets to validate their age and also annotated with gender and ethnicity where possible. Each user in our dataset was analyzed by two different annotators, and only those in which there was agreement for all characteristics were kept. Ultimately, we compiled a dataset of 1808 annotated Twitter profiles, and retrieved historical tweets from their accounts. Depending on individual usage patterns, we retrieved a very variable number of tweets. The minimum was 5, the maximum was 7115, the average was 226.6, the median was 96, and the standard deviation was 326.

Age Range	Training Count	Test Count
Generation Y	590	253
Generation X	352	152
Older	323	138

Table 1: Total counts of the annotated Twitter users in our training set and test set by age range.

For our first attempt to create an age prediction system, we attempted to use rules. However, we quickly found that even things like usage of currently trending slang were not reliable in predicting age groups. Moreover, rule based systems did not seem to have the potential to achieve even modest recall. Clearly, age prediction could not be accurately performed deterministically based on tweets, and a technique that used a complex evidence based model would be needed.

Our second attempt at age prediction then was to use popular machine learning text analytics models based on N-grams. We deployed these models using classifiers in the NLTK python package (Bird et al., 2009). We tried Naïve Bayes, and Maximum Entropy models for unigrams, bigrams, and trigrams. We found that it was optimal to require a minimum of 3 training corpus occurrences for an N-gram to be included in our feature space.

Description	GenY-F1	GenX-F1	Older-F1
1-Gram ME	69.5	17.3	38.4
2-Gram ME	69.5	17.3	38.4
3-Gram ME	66.6	13.0	24.5
1-Gram NB	73.0	36.1	18.3
2-Gram NB	73.0	36.1	18.3
3-Gram NB	72.6	33.2	23.6

Table 2: F1 scores by age range category for Naïve Bayes and Maximum Entropy unigram, bigram, and trigram models.

Table 2 depicts the test set results from our Maximum Entropy and Naïve Bayes analysis. Increasing the our granularity to include bigrams and trigrams resulted in an better training set performance for Maximum Entropy and Naïve Bayes, but those increases did not generalize to the test set. Maximum Entropy models saw degradation in accuracy with higher level N-grams. For Naïve Bayes, there was a slight improvement based on an increase in performance at predicting the oldest age range. Regardless, these results would not be suitable for a deployed system to make confident judgments.

As we began exploring other techniques which we will describe in more detail in subsequent sections, we use Paragraph Vector as provided by the original developers (Mesnil et al., 2015). Additionally, we used the theano-hf python package (Boulanger-Lewandowski et al., 2012) as the beginning building block for our deep learning based approaches.

3 Unsupervised Text Vectors

Neural Network Language Models (NNLMs) were first proposed by (Bengio et al., 2001), and have since become a major focus of research in building feature representations for text. (Mikolov et al., 2013), (Pennington et al., 2014), and (Levy and Goldberg, 2014) demonstrate that high quality vectors mapping N-gram phrases to latent vectors can be learned over large amounts of unlabelled data. These vectors have been shown to be able to naturally express synonymy through vector similarity and relationships through vector arithmetic. From a practical perspective, this work can be very useful to systems with limited training data as unlabelled public data is readily available, while supervised labeled training data often is not.

	Training				Testing			
Description	Accuracy	GenY-F1	GenX-F1	Older-F1	Accuracy	GenY-F1	GenX-F1	Older-F1
3-Gram ME	**87.3**	**88.1**	**82.9**	**90.0**	51.9	66.6	13.0	24.5
3-Gram NB	84.5	85.7	81.6	84.7	**56.4**	72.6	**33.2**	23.6
PV Logistic Regression	57.2	73.0	32.7	49.7	56.2	**72.8**	30.6	**46.9**

Table 3: Accuracies and F1 scores by age prediction category for Paragraph Vector (PV), Maximum Entropy (ME), and Naïve Bayes (NB) models.

In this paper we use Paragraph Vector, proposed by (Le & Mikolov, 2014), to build unsupervised language models. The key idea of this model is to predict nearby words with a fixed context window of surrounding words. Paragraph Vector extends to any segment of text with any length by allowing each unit of text (i.e. units in our experiments are a group of historical tweets for a particular user) to be represented by its own vector that is learned by contributing to the prediction of nearby words along with the words in the context window. Paragraph Vector has been shown to be a state of the art technique for analyzing supervised document level sentiment. However, we envision our end-to-end architecture as not being tied to a particular unsupervised feature learning technique. In fact, the drawback of Paragraph Vector is that all text units must be stored in memory, and an iterative inference step is needed during runtime. Eventually it is not unlikely that advances and variation in Recurrent Neural Network Language Models, as discussed in (Mikolov et al., 2010) and (Sutskever et al., 2011), or Recursive Neural Networks, as in (Socher et al., 2013) and (Socher et al., 2011), will provide a more scalable alternative for mapping text segments of arbitrary length to vectors.

In this section we will explore the performance of the unsupervised text vector component of our end-to-end architecture. We will first discuss the comparison between the unmodified Paragraph Vector method and popular N-gram machine learning models. Then we will discuss the effect of augmenting Paragraph Vector with unlabelled data.

3.1 Comparison with N-gram Models

In this experiment we implemented Naïve Bayes and Maximum Entropy N-gram models to serve as machine learning baselines over our age prediction dataset. We trained Paragraph Vector with a word context window of 8, 20 training epochs, and text vectors of length 300. After establishing text vectors for the training set of user tweet collections, we trained a logistic regression classifier as (Le & Mikolov, 2014) do in their original sentiment analysis paper.

Table 3 displays the results of this analysis. When it comes to testing accuracies and age range specific F1 scores, Paragraph Vector seems to result in the most well rounded representation, but the Naïve Bayes trigram model actually achieves a slightly higher overall accuracy. However, one clearly evident differentiator between the techniques can be seen in the breakdown of the results over the training set.

The trigram Maximum Entropy model experiences a 35.4% drop-off in accuracy from the training set to the test set, Naïve Bayes experiences a 28.1% drop-off in accuracy, and Paragraph Vector only falls 1%. It seems as though particularly for the case of the tougher Generation X and Older ranges, the N-gram models overfit on this small training set in a way that does not generalize. The Paragraph Vector model, however, has built a notion of text synonym that constricts its learning to knowledge that will generalize. Table 3 seems to indicate that despite similar performance, the Paragraph Vector model has a far better idea of its own true accuracy than N-gram models and has the potential at least to significantly improve whereas the N-gram models are much closer to their accuracy limitations given the small training dataset.

3.2 Knowledge Transfer From Unlabelled Data

In order to extend the Paragraph Vector model, we explored the possibility of expanding its knowledge coverage by incorporating unlabelled data.
As we were concerned about the effect on performance of both storing and conducting inference over text segment vectors at scale, we did not include any additional user profile vectors in our model. Instead, additional unlabelled tweets were added to the Paragraph Vector training and only the words were considered.

	Testing			
Logistic Regression	Accuracy	GenY-F1	GenX-F1	Older-F1
Age Corpus	56.2	72.8	30.6	46.9
Age Corpus + 1 Million Tweets	61.3	**78.2**	35.6	53.0
Age Corpus + 10 Million Tweets	**63.2**	77.9	**42.0**	**53.4**

Table 4: Logistic regression Paragraph Vector results with the incorporation of additional text.

Table 4 shows that as more tweets are included, the Paragraph Vector model becomes better. In fact, the addition of 10 million unlabelled tweets results in a 12.5% relative improvement in the accuracy of the original Paragraph Vector model. It should be noted that each of these corpuses was analyzed over 20 training epochs of Paragraph Vector. It is also important to note at this stage that we have found that the number of training epochs has a big impact on the quality of the text vectors produced by Paragraph Vector. The implication being that training on massive corpuses only makes sense if the time is allotted for a significant number of iterations.

4 Learning Deep Neural Networks from Unsupervised Text Feature Vectors

A logical first step in building powerful representations on top of unsupervised text vectors is to analyze the performance differences between logistic regression and generic deep neural network architectures. 3.03 million total free parameters is a good number that we established as the desired size for our neural network architecture. In these experiments (and all that follow) we kept that size constant across different numbers of hidden layers and every hidden layer was set to be the same size within an individual single task network.

We also restricted our analysis to Paragraph Vector with 10 million unlabelled tweets because it achieves the best performance with logistic regression. Our neural network leverages the Hessian Free Optimizer (Martens, 2010) and (Martens and Sutskever, 2011) in order to traverse pathological curvatures in the error function. We found this method to be considerably better than straightforward stochastic gradient descent in practice. Additionally, our deep neural network was initialized with greedy layer wise pretraining (Hinton et al., 2006). We used sigmoid activation units, a preconditioner, and a cross entropy loss function.

Our deep learning results are depicted in Table 5. Our network increase in performance as we increase the number of hidden layers until hitting a maximum total accuracy of 73.1% with three hidden layers. The three hidden layer network is the most efficient in its use of free parameters, and shines above the rest due to a considerable separation from the pack in predicting Generation X Twitter users – the toughest age range to predict.

		Testing			
# Hidden Layers	# Free Parameters	Accuracy	GenY-F1	GenX-F1	Older-F1
1	3.03M	71.1	87.3	46.4	63.6
2	3.03M	71.6	**88.2**	46.3	62.7
3	3.03M	**73.1**	87.5	**58.3**	66.2
4	3.03M	72.7	87.8	50.4	65.2
5	3.03M	72.0	87.1	46.7	**66.7**

Table 5: Results for different numbers of equal sized hidden layers with a fixed total parameter size.

5 Deep Multitask Learning Architectures

Multitask learning across deep neural network architectures is far from a new idea. The architecture portrayed in Figure 1, taken from (Socher and Manning, 2013), is seemingly of general consensus in the deep learning community (Bengio et al., 2013). The main idea is that a shared input is sent to an arbitrary amount of Neural Network hidden layers that are shared between related tasks and then classified by an arbitrary number of task specific Neural Network hidden layers and a task specific output layer.

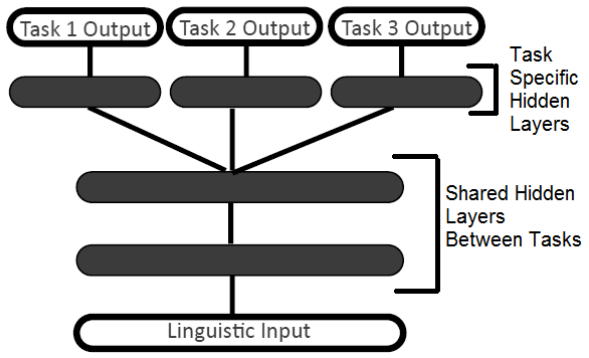

Figure 1: Standard Deep Multitask Learning Architecture Diagram

In (Collobert & Weston, 2008) this general architecture is extended in an attempt to perform Semantic Role Labeling and an unsupervised language model is used to initialize word vectors. However, it is important to note that they have

many more training examples in their experiments than we do. In a situation where there is a relatively small number of training examples, we believe it makes more sense to treat unsupervised text mappings as an input feature space for training that is shared across tasks (as opposed to just an initialized layer). Although feature spaces created by unsupervised learning could contain errors, with limited training examples algorithms cannot afford to perform sparse updates based on individual N-grams. It is imperative in learning relationships that generalize well and do not overfit to associate discoveries about phrases with synonyms and phrases of similar meaning. As we have already shown, doing this with high quality unsupervised feature vectors constrains the space of learning and prevents supervised machine learning algorithms from reading too much into misleading co-occurances present in smaller datasets.

A very simple paradigm of multi-task learning can be achieved by concatenating the output for each task and learning a single neural network that simultaneously classifies all tasks. Interestingly, this paradigm resulted in a consistent slight performance degradation in our experiments over single tasks. It seems like adding the extra output indicators must be complicating the process of minimizing error despite more information, even considering the small training corpus. Additionally, this method is only possible for training data that is jointly labeled, which significantly limits it applicability as a technique and seems inconsistent with the individual attention humans successfully exhibit when learning new skills. This limitation motivates the general architecture of Figure 1, which has no requirement for jointly labelled training data.

However, in the general multi-task deep learning model depicted in Figure 1, it is not clear how to approach the order of training tasks. In an extreme example, if you imagine first training one task for all epochs and then training another for all epochs, the first task would essentially serve as an initialization of the base network close to the input that will eventually get very much customized for the second task after enough iterations. We did not find this technique useful in our experiments. In fact, it seems like we may be relatively far from realizing the totality of the apparent promise of multi-task learning with an architecture in the form of Figure 1. In our experiments, we found that training a framework of that form by alternating between tasks every epoch (and even in mini-batches) actually resulted in a degradation of performance over single task learning. In fact, in (Collobert & Weston, 2008), where they loop through tasks in alternating order and update one random training example at a time, the authors find that Semantic Role Labeling is performed better over a large corpus just with Language Model initialization than it is with the additional contributions of knowledge of Part of Speech Tagging, Chunking, and Named Entity Recognition. This result is quite unintuitive given how related these tasks are, and points to a similar phenomenon to what we saw in implementing this paradigm.

6 Hidden Layer Sharing

Our proposed approach to multitask learning is performed with the following procedure:

1. Linguistic input is mapped to a shared unsupervised layer that serves as the effective input feature space for subsequent classifiers.
2. Each task is trained as its own deep neural network – the size of which is specified as a parameter of the model.
3. The output layer of each model is discarded and the top hidden layers for each model are concatenated.
4. The concatenated hidden layers are treated as a new input feature space to subsequent deep neural networks trained for each task. In our experiments we found a one layer logistic regression network with no additional hidden layers to make optimal use of free parameters, but this effect may change for different domains.

Figure 2 depicts an example architecture for hidden layer sharing between two tasks. In contrast to Figure 1, Figure 2c only illustrates classification of a single output at a time. This serves to underscore a critical practical point about prioritization.

In practice the number of free parameters is a constrained value for a production NLP system. We expect machine learning models to increase in performance with an increase in free parameters. On the other hand, there are practical limits

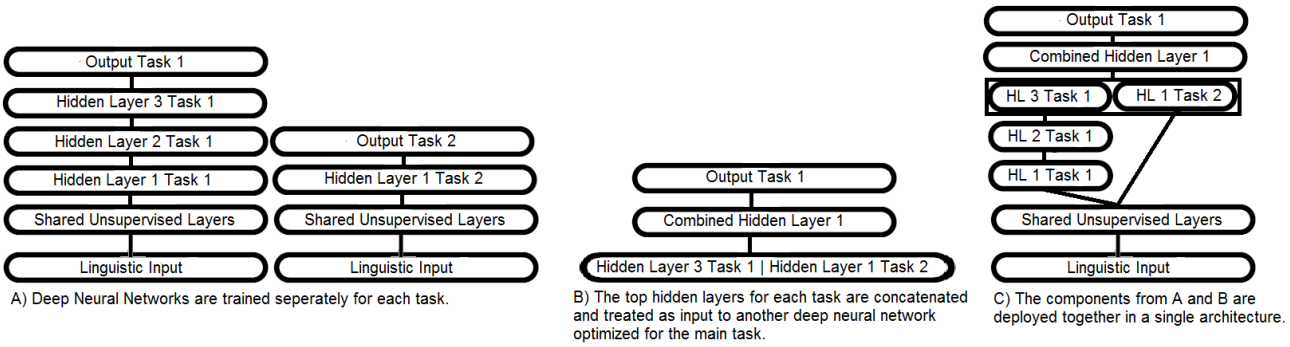

Figure 2: An example of the process and final deployment architecture for our hidden layer sharing approach. Task 1 is the main task to optimize. In this configuration, Task 1 is allotted three task specific hidden layers, Task 2 is allotted one task specific hidden layer, and the network that processes the output from the combined hidden layers in A is allotted one hidden layer on top of the combined input. The logical flow of steps goes from A to B for training and C for deployment. Optionally, fine-tuning can be conducted with the architecture in C.

imposed by the direct relationship between increasing the number of free parameters, increasing a model's memory footprint, and decreasing its runtime throughput. That being said, given the modern hardware these systems are deployed on today, most models hit a point of diminishing returns where increasing parameters has less and less impact on the model's accuracy. As such, the practical promise of multitask learning and knowledge transfer techniques today is to achieve a lift in predictive performance of models while staying constant at the allowable limit for total free parameters. When viewed in this way, it is clear that when considered as the main task being optimized, Task 1 would probably benefit from a different split of free parameters than Task 2 in Figure 2. All else being equal, although Task 2 is useful for improving Task 1, it is not as mission critical as the main task, so Task 1 likely should have more dedicated free parameters than Task 2 if you are classifying Task 1. The reverse would be true if you were classifying Task 2.

In our experiments we see two major positive effects of the hidden layer sharing technique. First, training the models separately seems to allow for a more stable learning for each task that overcomes early local minimums hit by other common architecture types. Second, the ability to directly specify the number of free parameters allocated to each model in early layers results in an ability to tune models for optimal prioritization of related tasks.

7 Measuring The Effectiveness of Hidden Layer Sharing

As discussed in Section 6, a key aspect of our hidden layer sharing approach is the ability to directly adjust the prioritization of tasks. For the case of training age prediction alongside the gender prediction task, we saw significant gains by limiting the amount of parameters in the model allocated to gender prediction. Training the model with a 50-50 split in free parameters allocated between tasks resulted in 68.9% total accuracy (a net decrease in performance from single task results), however, a 70-30 split in favor of the age prediction task brought total accuracy to 73.3%. A 90-10 split achieved the best two task result with 75.0% total accuracy. For the case of training age prediction alongside the ethnicity prediction task, we saw the opposite relationship. When the ethnicity learning task wasn't given enough stake in the shared hidden layer at a 90-10 free parameter split, it hurt our predictive accuracy by bringing it down to 70.7%. However, at an even 50-50 split the ethnicity task free parameters helped age prediction learning enough to overcome our 3 hidden layer single task result by achieving 73.9% total accuracy.

	Testing			
Description	Accuracy	GenY-F1	GenX-F1	Older-F1
3-Gram NB	56.4	72.6	33.2	23.6
PV Logistic Regression	56.2	72.8	30.6	46.9
PV Logistic Regression + 10M Tweets	63.2	77.9	42.0	53.4
3 Hidden Layer NN	73.1	87.5	58.3	66.2
3 Hidden Layer + Gender	75.0	87.6	58.3	**66.9**
3 Hidden Layer + Gender + Ethnicity	**75.9**	**89.4**	**59.9**	62.4

Table 6: Top results with a constrained free parameter size at different architecture points.

Table 6 highlights our best result, which came from integrating a scaled down version of the three hidden layer model with enough free parameters left over to give ethnicity and gender each an equal 10% of the total free parameter stake in the model. A logistic regression layer was built on top of the concatenated shared hidden layers to create a final output. 75.9% total accuracy constitutes a 3.8% relative improvement over deep learning models

due to knowledge transfer from two related tasks and a 34.6% relative accuracy improvement over the best performing baseline N-gram model. The hidden layer sharing approach was capable of integrating both gender and ethnicity detection as related tasks to age detection for significant additional gains on top of the large gains resulting from building deep learning on top of unsupervised language model feature vectors. This is a phenomenon that was expected, but not achieved with concatenated output and joint learning driven shared hidden layer architectures.

8 Extensibility of Architecture to Incorporate Available Metadata

Details	# Free Parameters	Accuracy	GenY-F1	GenX-F1	Older-F1
1 Hidden Layer	3.03M	71.1	87.3	46.4	63.6
3 Hidden Layers	3.03 M	**73.1**	87.5	**58.3**	66.2
1 Hidden Layer with Gender	3.03M	81.0	88.2	64.7	82.8
3 Hidden Layers with Gender	3.03M	86.6	88.0	76.4	93.9
1 Hidden Layer with Gender and Ethnicity	3.03M	88.6	89.5	78.1	**97.5**
3 Hidden Layers with Gender and Ethnicity	3.03M	**90.1**	**89.6**	**82.4**	**97.5**

Table 7: Comparison of results in age range prediction between neural network architectures with a fixed parameter size that are given gender and ethnicity information as structured metadata.

Beyond being able to leverage knowledge from multiple related learned tasks, it is important for a social media analytics solution to be able to properly leverage structured metadata when available. To showcase the ease in which a model in our architecture could do this, we ran an experiment assuming that gender and ethnicity are always given as metadata to our system. In this case we can see in Table 7 that both a single hidden layer model and multiple hidden layer models can benefit significantly from additional structured input that is concatenated with the input unsupervised language model feature vectors. Our 3 hidden layer model from before is able to efficiently incorporate in this structured data for 23.2% relative improvement over the same single task model. This is a very encouraging result to achieve 90.1% total accuracy with such a small age related training set.

The 14.2% gap between our hidden layer sharing result and what is possible with direct knowledge of the same tasks as metadata points out that if we had more training data on related tasks such as gender and ethnicity, it should be possible to achieve high accuracy results without the need for the metadata being directly given. Limitations in accuracy increases resulting from knowledge transfer are at least in part due to the limited accuracy for the individual gender and ethnicity tasks in our current experiments, which are learned over the same small dataset used for age prediction.

9 Conclusion and Future Work

Prediction tasks like age prediction based solely on historical tweets from a user are not possible using rule based techniques and are not possible with limited training data for N-gram machine learning techniques. In this paper, we have shown that using modern machine learning techniques such as the addition of unlabelled training data, deep learning, and knowledge transfer between related tasks, it is possible to achieve 75.9% predictive accuracy with limited training data. In fact, we have shown that these models are very extensible and achieve 86.6% accuracy for the common case where gender is known. Moreover, we can achieve 90.1% predictive accuracy when other useful metadata like ethnicity is present.

In this paper we have proposed a text analytics process flow and hidden layer sharing architecture suitable for solving tough prediction problems on noisy social media text. However, our approach in this paper can be translated to other even seemingly unrelated domains as well, such as business to business lead prediction, which will be the focus of future publications. Our hidden layer sharing approach gives developers the power to specify how a deep neural network stores and prioritizes knowledge between related tasks, where popular techniques generally allow the neural network to figure this out.

The success of this approach points out the need for improvement of shared hidden layer deep neural network approaches which in some cases have a difficult time prioritizing effectively and balancing learning across multiple complex error functions. Additionally, the huge improvements we see with direct knowledge of structured metadata are indicative of the potential that multitask architectures have for classification problems in the social media domain with limited training data.

References

Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. 2001. A Neural Probabilistic Language Model. *Advances in Neural Information Processing Systems '2000,* pages 932-938.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 35, no. 8, pages 1798-1828.

Steven Bird, Edward Loper, and Ewan Klein. 2009. Natural Language Processing with Python. *O'Reilly Media Inc.*

N. Boulanger-Lewandowski, Y. Bengio and P. Vincent. 2012. Modeling Temporal Dependencies in High-Dimensional Sequences: Application to Polyphonic Music Generation and Transcription. *In Proceedings of ICML*, page 29.

Ronan Collobert, and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160-167.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, pages 1-12.

Geoffrey Hinton, Simon Osindero, and Yee-Whye Teh. 2006. A fast learning algorithm for deep belief nets. *Neural computation* 18, no. 7, pages 1527-1554.

Nadin Kökciyan, Arda Celebi, Arzucan Ozgür, and Suzan Usküdarlı. 2013. Bounce: Sentiment classification in Twitter using rich feature sets. In *Second Joint Conference on Lexical and Computational Semantics (* SEM)*, vol. 2, pages 554-561.

Rajasekar Krishnamurthy, Yunyao Li, Sriram Raghavan, Frederick Reiss, Shivakumar Vaithyanathan, and Huaiyu Zhu. 2009. SystemT: a system for declarative information extraction. *ACM SIGMOD Record* 37, no. 4, pages 7-13.

Quoc Le, and Tomas Mikolov. Distributed Representations of Sentences and Documents. 2014. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1188-1196.

Omer Levy, and Yoav Goldberg. Neural word embedding as implicit matrix factorization. 2014. In *Advances in Neural Information Processing Systems*, pages 2177-2185.

James Martens. Deep learning via Hessian-free optimization. 2010. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 735-742.

James Martens, and Ilya Sutskever. 2011. Learning recurrent neural networks with hessian-free optimization. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1033-1040.

Grégoire Mesnil, Tomas Mikolov, Marc'Aurelio and Yoshua Bengio. 2015. Ensemble of Generative and Discriminative Techniques for Sentiment Analysis of Movie Reviews. Submitted to the *workshop track of ICLR 2015*.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernocký, and Sanjeev Khudanpur. Recurrent neural network based language model. 2010. In *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September 26-30, 2010*, pages 1045-1048.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. 2013. In *Advances in Neural Information Processing Systems*, pages 3111-3119.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014)* page 12.

Michael Speriosu, Nikita Sudan, Sid Upadhyay, and Jason Baldridge. 2011. Twitter polarity classification with label propagation over lexical links and the follower graph. In *Proceedings of the First workshop on Unsupervised Learning in NLP*, pages 53-63. Association for Computational Linguistics.

Richard Socher, Christopher D. Manning, and Andrew Y. Ng. Learning continuous phrase representations and syntactic parsing with recursive neural networks. 2010. In *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*, pages 1-9.

Richard Socher, and Chrisopher Manning. 2013. Deep Learning for Natural Language Processing (without Magic). 2013. Tutorial at *NAACL HLT 2013*.

Richard Socher, Alex Perelygin, Jean Y. Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, vol. 1631, pages 1642.

Ilya Sutskever, James Martens, and Geoffrey E. Hinton. 2011. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1017-1024.