Natural Language Technology to Ensure the Safety of Speech Information

Ievgen Iosifov[1,2], Olena Iosifova[1], Volodymyr Sokolov[2], Pavlo Skladannyi[3], and Igor Sukaylo[4]

[1] *Ender Turing OÜ, ½ Padriku str., Tallinn, 11912, Estonia*
[2] *Borys Grinchenko Kyiv University, 18/2 Bulvarno-Kudriavska str., Kyiv, 04053, Ukraine*

Abstract

This paper is focused on Natural Language Processing (NLP) and speech area, describes the most prominent approaches and techniques, provides requirements to datasets for text and speech model training, compares major toolkits and techniques, and describes trends for NLP and speech domain.

Keywords

Neural network, natural language technology, natural language processing, automatic speech recognition, deep learning, encoder, decoder, word embedding, hidden Markov model.

1. Introduction

Significant advances in Deep Learning (DL) of the last decade uncover new possibilities and demands for businesses, governments, and citizens. Such advances in Natural Language Technology (NLT) allow businesses to automate most routine and boring tasks in communication with customers and direct people's minds to more exciting and creative tasks [001].

To fully leverage NLT, two central stacks of technologies should be combined:

- Speech technologies—to translate speech into text and vice versa.
- NLP—to understand, interpret, and generate information in a text.

This work reviews existing knowledge, directions, and avenues for future research in this increasingly important domain/area NLT/NLP.

NLP is a field of artificial intelligence that helps the computer to understand and generate text. NLP is broadly used in many tasks: dialogue systems, sentiment analysis, machine translation, information retrieval, summarization, question answering, etc. During the last decade, there were few breakthroughs in DL fields, first for image recognition and later for natural language, which attracted researchers and businesses' colossal interest. We will review the most prominent and fundamental techniques, which significantly improve machine skills in natural language: Recurrent Neural Networks (RNNs), embedding concept, the concept of decoder and encoder, and shortly attention and transformers. Without this technique, it is hard to imagine such interest in the NLP field.

Automatic Speech Recognition (ASR) and speech generation are techniques to convert human speech to text and back. After a structured communication system called language evolved, speech is the main instrument of any communication between human beings. For machines, such language is digits, and it was many iterations to present the human speech to machine understandable language.

Sect. 2 will review the latest and most promising techniques, like the hybrid Hidden Markov Model (HMM) and end-to-end systems, combined with Deep Neural Networks (DNNs). Additionally, we will review data requirements in Sect. 3. As with currently available frameworks, input data quality and relevance contribute a major if not overwhelming part of the resulting model quality. In Sect. 4, a

CPITS-II-2021: Cybersecurity Providing in Information and Telecommunication Systems, October 26, 2021, Kyiv, Ukraine
EMAIL: ei@enderturing.com (I. Iosifov); oi@enderturing.com (O. Iosifova); v.sokolov@kubg.edu.ua (V. Sokolov); p.skladannyi@kubg.edu.ua (P. Skladannyi); i.sukailo.asp@kubg.edu.ua (I. Sukaylo)
ORCID: 0000-0001-6203-9945 (I. Iosifov); 0000-0001-6507-0761 (O. Iosifova); 0000-0002-9349-7946 (V. Sokolov); 0000-0002-7775-6039 (P. Skladannyi); 0000-0003-1608-3149 (I. Sukaylo)

comparative analysis of approaches and frameworks is presented. Evaluation metrics to measure NLP systems and ASR presented in Sect. 5, end-to-end training approaches in Sect. 6, and trends in Sect. 7.

2. High-Level Overview of Natural Language Technology

2.1. Natural Language Processing Techniques Review

Below we will review the main breakthrough points in the NLP area of the last decade. We will start with RNNs as the central concept in NLP (recurrent and combining information from previous iterations), then will present more advanced techniques to do feature engineering (not just a one-hot encoding of words in the dataset, but complicated vector representation with context and additional information related to word).

As the main starting point of the NLP, the area was machine translation. It is natural that the concept of encoder-decoder and sequence-to-sequence evolved, which will be covered as well. Attention and transformer will be reviewed as the last achievements in the NLP area.

2.1.1. Recurrent Neural Networks

RNNs were the primary building block for NLP tasks for an extended period. The main difference of RNNs from other DL architectures is the ability to remember data for sequence, not only for the last cell (word/token).

The network takes **X** as the input vector (usually encoded-word representations) and produces **Y** as the output vector. Each RNN cell takes current input x_t and previous hidden state (activation) $h_t - 1$, which stores information extracted during previous iterations. The network learns weights (parameters) *Wh*, W_x, and bias b_a through the weights learning process. At each iteration of forwarding propagation, non-linear activation function g such as *tanh* (or similar) is applied to calculate output hidden state (activation)

$$h_t = g(W_h\, h_{t-1} + W_x\, x_t + b_a). \tag{1}$$

Additionally, *softmax* (activation function *g*) may be applied in the end if output predictions needed by the task

$$y_t = g(W_y\, h_t + b_y). \tag{2}$$

Most important for tasks in the NLP area is that output includes information from previous, not only last. It is crucial mainly because of the nature of language. One last word (token) is not enough to understand the context of the sentence. Such a type of connection is called a recurrent connection (see Fig. 1).

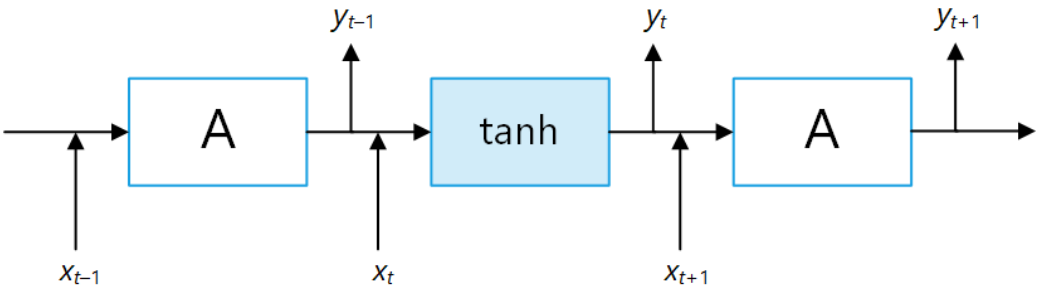


Figure 1: RNN design

This idea and concept of recurrent connection and context significantly affect the current state of the NLP area. RNNs have many disadvantages, though, like unidirectionality, the problem with capturing mid and long-term connections/dependencies inside a sequence. Today it is rare to find RNN as the underlying architecture. More complicated architectures arrived

based on RNN, like Gated Recurrent Units (GRU), Long Short-Term Memory (LSTM), and some architectures come to NLP from image recognition, like Convolutional Neural Networks (CNNs) [1].

2.1.2. Word and Contextual Embedding Approaches

The primary purpose of embedding is to represent tokens (documents, phrases, context, a piece of a word, or a character) as a numerical vector. Then neural networks can calculate and use the probability distributions or likelihoods to separate semantically similar categories. So that different tokens with similar meanings will have closer vectors and different by meaning groups of tokens can be separated in vector space. [4] made famous the idea that "a word is characterized by the company it keeps." Lately, new approaches have appeared. Contextual embedding [5] is a representation of a token within its context. During the embedding processes, information of a token presence in different contexts is considered [6].

2.1.3. High-Level Encoder-Decoder Architecture

The encoder-decoder approach became a breakthrough and led to a significant increase in the performance of language models. The input sequence ["What's"|"up"|"?"] at the embedding layer gets numerical representation. Then numerical representation is sequentially fed to the RNN. After all, inputs get through RNN produces output. This part is called an encoder—it encodes input sequences (see Fig. 2).

The result of the encoder transfers to the decoder. The decoder generates predictions of a resulting sequence until it gets to the *end of the sentence token*.

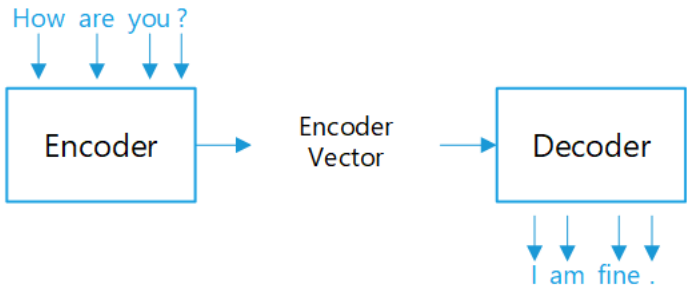


Figure 2: An example of an encoder-decoder architecture

The most outstanding achievement of encoder-decoder is the possibility to use it to create end-to-end models correctly and the possibility to handle input and output sequences of different lengths. The problem of inconsistent input and output lengths is especially actual in neural machine translation.

The encoder-decoder architecture is usually based on two RNNs or LSTM. The encoder encodes all input sequences and stores all information in the encoder vector, and the Decoder creates result predictions.

2.1.4. Attention

The main limitation of RNNs is dependencies tracking in long sentences. Long sentences (more than 20 words) just can't be stored effectively in the output vector of RNN. That is why researchers come up with the attention mechanism.

The idea of attention is the same as attention from the human reading process. For humans, a few words from a sentence are enough to understand the sentence well. During the translation process, humans need just a few main words to translate, all other words simply out of attention. The same for attention: the decoder focuses on some particular part of the source at each step. Decoder focuses only on particular words at each step (increased saturation represents more attention), not the whole input sequence. The attention mechanism uncovers such possibility to a decoder by attention weights and context vector [14].

2.1.5. Transformers

The transformer is one of the last breakthroughs which accelerates NLP significantly. The transformer is architecture builds on top of the encoder-decoder concept and is hugely based on the attention concept. The main breakthrough was parallelization by replacing sequential computation (RNNs or CNNs) with an attention-based network. The main components and concepts of this architecture will be presented and described below.

The *encoder* consists of multiple stacked self-attention and feed-forward layers with residual connections and a positional encoder. The embedding layer is usually applied at the bottom to convert the input sequence to numerical representation. The feed-forward network does not have dependencies and thus can be parallelized. That is an essential concept behind the Transformer's possibility to learn on a vast amount of data that LSTM and GRU can't afford.

The *decoder* also consists of multiple (equal to the encoder) stacked self-attention, feed-forward layers with residual connections, and additionally encoder-decoder attention layer in the middle. In comparison to the encoder, the decoder's self-attention layer differs. The main idea here is masking future positions. In the encoder, each position can attend to all positions. Still, in the decoder, to prevent leftward information flow to preserve the auto-regressive property, each position can attend only to early positions in the output sequence [13].

2.2. Speech Techniques Review

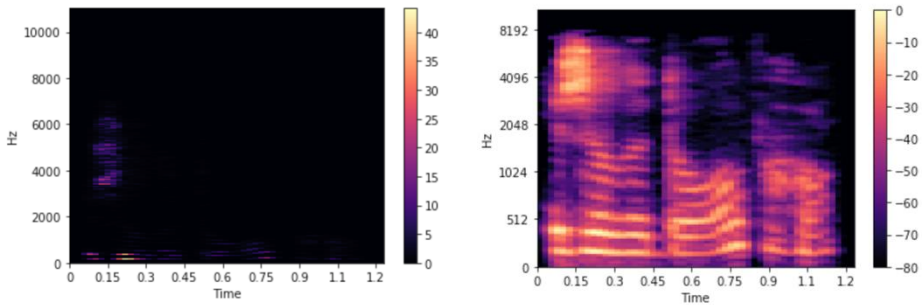
The main goal of speech systems is to convert input audio wave sequence into text representation in the case of an ASR system and vice versa in the case of speech generation from the text (text-to-speech, TTS) [18]. There are two main approaches here:

- Hybrid models based on HMM-DNN ASR systems.
- End-to-end ASR systems.

2.2.1. Feature Representation

To represent the input audio sequence in format machine understand, we have to ally some transformations. Research shows that it is not good enough to convert input waves into digits of corresponding amplitudes by sampling audio signals. Such features are very uninformative for the training process to squeeze as much information as possible to remember and generalize the audio signal.

The spectrogram was obtained during Fast Fourier Transformation (FFT) to represent time (or similar non-linear), frequency, and energy at each point of time. Trans-formation represents features in acoustic frames format (20–40 ms). Mel-Frequency Cepstral Coefficients (MFCCs) or perceptual linear prediction is a common choice of non-linear transformation techniques for feature extraction for ASR data [19]. The classical FFT is not suitable for determining patterns (see Fig. 3).



(a) (b)

Figure 3: Spectrogram of (a) FFT and (b) MFCC

2.2.2. Hidden Markov Model

To reconstruct the utterance that has just been said by putting the correct phonemes after HMM was used. It is done by using statistical probabilities that one phoneme follows the other.

In simplified words, HMM consists of three different layers:

1. The heart of the HMM model is an acoustic model that checks on the acoustic level the probability that the phoneme it recognizes is that phoneme.

2. After that lexicon (pronunciation) model is applied, checking the probability that recognized phonemes next to each other can stand next to each other.

3. In the end, the language model applied (usually in the way of *n*-grams) check on the word level and whether words standing next to each other make sense. As an example here, the model will choose "cat paws" instead of "cat pause" [21].

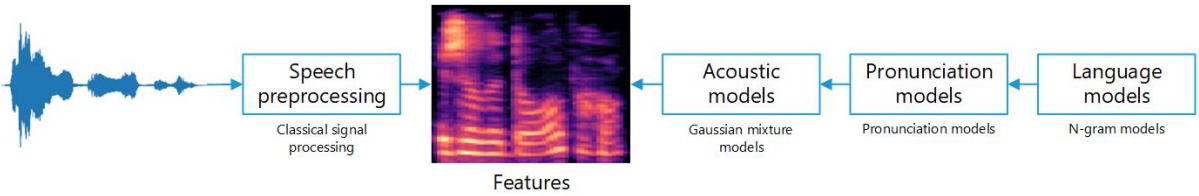


Figure 4: Hybrid three-level ASR

2.2.3. End-to-End Model

As can be seen from above—one of the significant limitations of HMM models is phoneme to grapheme mapping. Especially this problem arrived for a low-resource language where no one tried to prepare such a model. It might be very time-consuming to the prepared dataset for such mapping. That is one reason that simplified end-to-end models evolved. The main inspiration was to train the model with as few labeling and intermediate steps as possible. The model should learn by themselves map phonemes to grapheme directly or indirectly using the same input data used for current training. Another motivation is to jump into an area of unsupervised training, to exploit the vast amount of unlabeled audio data stored on the Internet.

There are few varieties of end-to-end ASR systems architectures. At the same time, all of them are built on two types: Connectionist Temporal Classification (CTC) and sequence-to-sequence (encoder-decoder-based).

Before CTC main limitation of the end-to-end ASR system was that the model needed to have the whole sentence to start the translation. It means no streaming decoding possibility. CTC map input sequence **X** (MFCC) to output sequence **Y** (letters).

One of the CTC breakthroughs is that kind of local attention introduced, which splits the continuous speech, and then the current modeling unit using attention runs on each split segment. By doing this, the whole utterance is split into small segments, and local attention is used to predict features (letter) [26].

2.3. Text-to-Speech

Standard TTS systems consist of a few parts:

- RNN modifications (LSTM, GRU, etc.) for recurrent sequence-to-sequence feature prediction network that maps character embedding to MFCC spectrograms (description of these components is familiar very similar to above-described components).
- Vocoder system that synthesizes waveforms from those spectrograms. The relationship between linguistic features and vocoder parameters that represent the vocal cord and vocal tract characteristics learned by acoustic models. Vocoder parameters are generated (in the synthesis stage) from the trained acoustic models, and a speech waveform is synthesized using high-quality vocoder systems [31].

3. Data Requirements

Data preparation is a significant step in any DL area, and NLP is not an exclusion. As DL techniques are remembering and generalization of training datasets, it is hard to overestimate the impact of a good and bad quality dataset. As in any other DL task, data is presented in features and relevant label form.

3.1. Natural Language Processing Data Requirements

Requirements for input data are the same as for other tasks in the DL area: input data should be as much as possible closer to the domain model will work in. In simple words, you can't train a model for medical anamneses prediction using a financial data dataset. If the training process model didn't see examples of token/word in the dataset, it will just not react to it. Over the past years, huge progress has been made to overcome such limitations, and embedding techniques with pre-trained token embedding helps a lot. Still, it is hard to overemphasize how better the trained model would be if you use relevant training data.

The NLP area has some specific requirements: data should be split by sentences, which is a big problem for ASR. For such specific NLP tasks as punctuation, the typical case is to generate a synthetic dataset and label it in the most suitable task way [35].

Nowadays, there are many labeled and much more unlabeled datasets, which is a good starting point for significant tasks, so software engineers do not need to collect and label datasets on their own.

3.2. Speech Data Requirements

Datasets for ASR are some audio files (usually, 3–20) and related text transcripts (labels). All digits should be denormalized to a text representation. For the model to get used to acoustic, were four and format sound almost the same, and if in training dataset it will be "4" and "format," it will be much harder for the model to generalize acoustically. Such a denormalization task can be very complicated. Let's imagine number "3," which may represent "three," "third," etc. And such a task is even more complicated for non-English languages.

The above gives us some information on how hard to prepare a good dataset for ASR, especially for low resource languages. There are few directions to overcome such limitations, for example, (1) use non-labeled audio data (unsupervised learning), which arise new limitations of computational resources; (2) to generate such dataset iteratively using smaller to generate bigger where we have generated 2,500 hours dataset using just 100 hours as a starting point. As an example of language to test such automated ASR dataset generation pipeline, we took low resource Ukrainian language, which is very limited in terms of available ASR datasets (see Table 1).

Table 1
Datasets and sources for Ukrainian speech corpus

Dataset name	Class	Duration, hours	Quality
Ukrainian Corpus for Broadcast Speech [37]	ASR dataset	366	—
Multi-speaker Corpus "UkReco" [39]	ASR dataset	—	—
M-AILABS Ukrainian Corpus [40]	Books	87	Excellent
Ministry of Education, Culture and Science Lessons [41]	YouTube	29	Good
Deutsche Welle in Ukrainian [42]	YouTube	70	Ordinary
Telebachennya Toronto [43]	YouTube	60	Ordinary
Mozilla Common Voice [44]	Mozilla	22	Excellent
TEDx Talks [45]	TEDx	<50	Good

One of the main difficulties in data preparation for ASR is that many audios are stored in mono format, with mixed channels and hence mixed speakers. There are a few techniques to separate such audios by using voice activity detection and several speaker identifications.

ASR systems are pretty demanding for data. You need approximately the following amount of data to train the good model:

- Five thousand hours for a hybrid approach.
- Ten thousand hours for an end-to-end approach.
- Thirty thousand hours for an unsupervised learning approach.

4. Models and Frameworks

4.1. Natural Language Processing Models

There are a few main divisions of approaches used to solve NLP tasks. Main are either to use pre-trained models like (BERT [46], RoBERTa [47]) or to train a model from scratch based on Bidirectional RNN (BRNN), LSTM, CNN architecture (see Fig. 5).

Pre-trained models can be divided into encoder-based and decoder-based. We will not review non Transformer based pre-trained models as with distilled pre-trained Transformer based models, you can achieve almost the same model efficiency, like BRNN but with significantly higher accuracy.

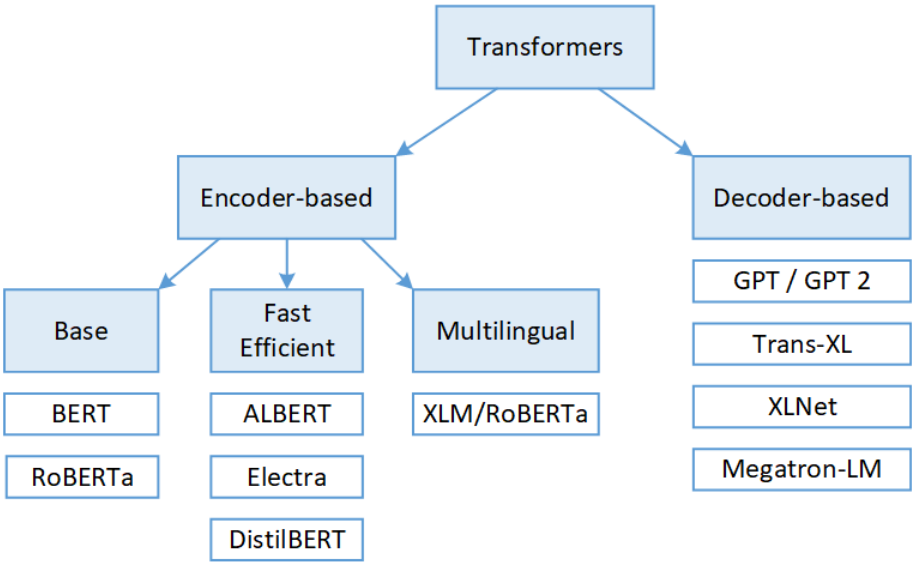


Figure 5: Model classification

If the task or domain is specific and you can't exploit the pre-trained model, it is good to train the BRNN, LSTM model from scratch (see Fig. 6).

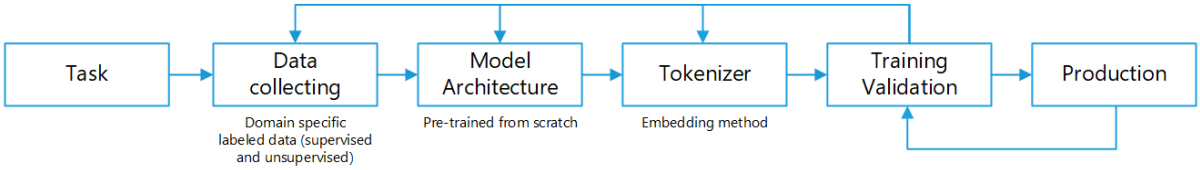


Figure 6: Model training process

4.2. Speech Processing Toolkits

A comparative analysis of a conventional Speech Processing Toolkit (SPT) and tools to build an ASR model is presented in Table 2. We compared in a few dimensions, and the most important ones are demand in data amount to train or fine-tune your model and learning curve to start working with the toolkit.

Table 2
Comparative analysis of existing software

Product	Type	Training type	Data amount	Learning curve
Kaldi [56]	Hybrid SPT	Supervised	Medium-low	Hard
Julius [57]	Hybrid SPT	Supervised	Medium-low	Hard
DeepSpeech [58]	End-to-end SPT	Supervised	High-medium	Easy
EspNet [59]	End-to-end SPT	Supervised	High-medium	Easy
FairSeq [60]	End-to-end SPT	Unsupervised	Medium-low	Normal

As can be seen from comparative analysis, no one toolkit can handle all steps of ASR data-preprocessing (including collection, splitting, labeling, preparing for ASR expected format), and model training. That is why we believe more teams contribute to creating more frameworks and toolkits to lower the learning curve and demand in training data.

The results of speech recognition make it possible to automatically track the illegal activity of users of the information system using keywords [002].

5. Evaluation Metrics

5.1. Natural Language Processing Metrics

Because the NLP area is quite broad and the number of tasks in NLP is vast—there are no standard metrics for all tasks. We can split metrics by clustering tasks and highlight next:

- Machine translation models: bilingual evaluation understudy is a performance metric to measure the performance of machine translation models. It evaluates how well a model translates from one language to another.
- Language understanding evaluation: general language understanding evaluation is a benchmark based on different types of tasks rather than evaluating a single task. The three major categories of tasks are single-sentence tasks, similarity and paraphrase tasks, and inference tasks.

At the same time, there are many fine-tuning tasks, like part of sentence tagging, named entity recognition, etc. In such tasks, the most common metric is to count accuracy through *precision*

$$P = N_{tp} / (N_{tp} + N_{fp}) \tag{3}$$

where N_{tp} is a true positive, N_{fp} is a false positive.

And *recall* the equality calculates the coefficient

$$R = N_{tp} / (N_{tp} + N_{fn}) \tag{4}$$

where N_{fn} is a false negative.

F1 Score calculates from (1) and (2) [62]:

$$F1 = 2 \cdot P \cdot R / (P + R). \tag{5}$$

5.2. Speech Processing Measurement Criteria

It is much easier to come up with a unified metric for ASR than for NLP tasks, as we only have to measure if the word is recognized correctly or not. Hence, Word Error Rate (WER) is the most common to see an ASR accuracy metric. The lower WER is better than the ASR system. WER can then be computed as:

$$WER = (N_S + N_D + N_I) / (N_s + N_D + N_C) \tag{6}$$

where N_S is the number of substitutions, N_D is the number of deletions, N_I is the number of insertions, N_C is the number of correct words [63].

It is worth mentioning that WER is very sensitive to domain and acoustic. For example, low (good) WER of 5% for literature (trained on books) domain can have 20–30% WER for call center calls [64].

6. End-to-End Training Approaches

With the mentioned breakthrough in NLP and speech areas, more and more businesses see great opportunities for the implementation of NLT based systems. That drives demand for NLP / speech engineers. Such demand can't be satisfied with the existing supply, so the learning curve should be lowered. And one of the significant steps to such supply increase is end-to-end systems, which is much simpler from the end-user (engineer perspective). We see a huge trend and opportunity in end-to-end training approaches. Yes, it still requires more data, is less effective, etc., but we believe it is the future of NLP and speech systems.

Data labeling is the most comprehensive, long, and expensive process—last year's trends are to use unlabeled data. It is much easier to collect data than to collect labeled data specifically for your domain

or low-resource language. However, even unlabeled data have to comply with requirements. Today's unsupervised learning approaches for NLP became standard, and with approaches for distillation and pruning on top of base training, they become effective and practical. We are still looking forward to more effective unsupervised learning in the speech area that will require not 50,000 hours of speech and enormous resources to train the model but more practical.

To teach the model to generalize using unlabeled data demands a much more significant amount of data and is thus very demanding in computational resources (50,000 hours of unlabeled data to train a state-of-the-art ASR model). That is why the third trend is the pre-trained models. For NLP, it is common to train a model on a significant amount of data once and then fine-tune for underlying tasks. As an example mentioned before—using a pre-trained BERT model to mark punctuation after ASR for NLP processing. Not to mention how huge a lower learning curve and time investments are for an engineer to prepare a production-ready model. You do not need to collect vast datasets only to train, for example, tokenizer. For speech, pre-training is a common cause for hybrid models and still have to be evolved for end-to-end approaches (especially for unsupervised learning), as it is clear that researcher or software engineer will not have the possibility to spend hundreds of thousands of USD to train the model using unsupervised data.

7. Conclusions

As humans learn many languages to understand other humans (even within one country), we expect to see more advanced multilingual models that can accurately understand multilingual dialogues.

We believe all these trends are possible because great-specialized frameworks for NLP and speech appeared, as a few examples: HuggingFace [65] for NLP, Kaldi [56], ESPnet [59], FairSeq [61] for speech-to-text recognition, Tacotron [31] for TTS synthesis. We expect to see advances in this area of Frameworks and toolkits, gathering last achievements and preparing interfaces to use them for more and more engineers.

8. References

[1] S. Gnatyuk, et al., Method of cybersecurity level determining for the critical information infrastructure of the state, in: 2nd International Workshop on Control, Optimisation and Analytical Processing of Social Networks (2020) 332–341.

[2] O. Iosifova, et al., Techniques Comparison for Natural Language Processing, in: Proceedings of the Modern Machine Learning Technologies and Data Science Workshop 2631 (2020) 57–67.

[3] J. R. Firth, in: A Synopsis of Linguistic Theory (1957) 1930–1955.

[4] M. Peters, et al., Deep Contextualized Word Representations, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies; Association for Computational Linguistics: New Orleans, Louisiana 1 (2018) 2227–2237.

[5] Q. Liu, M. J. Kusner, P. Blunsom, A Survey on Contextual Embeddings (2020) 1–13. arxiv:200307278.

[6] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, in: MIT Press (2016) 462–480.

[7] D. Bahdanau, K. Cho, Y. Bengio, Neural Machine Translation by Jointly Learning to Align and Translate 2016, 1–15. arxiv:14090473.

[8] K. Cho, et al., Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing; Association for Computational Linguistics: Doha, Qatar (2014) 1724–1734.

[9] G. Hinton, et al. Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups, IEEE Signal Process. Mag. 29 (2012) 82–97. doi:10.1109/MSP.2012.2205597.

[10] L. R. Rabiner, B. H. Juang, Fundamentals of Speech Recognition, in: PTR Prentice Hall (1993) 342–368.

[11] L. E. Baum, T. Petrie, Statistical Inference for Probabilistic Functions of Finite State Markov Chains, Ann. Math. Stat. 37 (1966) 1554–1563. doi:10.1214/aoms/1177699147.

[12] Y. Wang, et al., Tacotron: Towards End-to-End Speech Synthesis (2017) 1–10. arxiv:170310135.
[13] I. Iosifov, O. Iosifova, V. Sokolov, Sentence Segmentation from Unformatted Text Using Language Modeling and Sequence Labeling Approaches, in: Proceedings of the 2020 IEEE International Scientific and Practical Conference Problems of Infocommunications. Science and Technology; IEEE: Kharkiv, Ukraine (2020) 335–337. doi:10.1109/PICST51311.2020.9468084.
[14] O. Romanovskyi, et al., Automated Pipeline for Training Dataset Creation from Unlabeled Audios for Automatic Speech Recognition, Advances in Computer Science for Engineering and Education IV 83 (2021) 25–36. doi:10.1007/978-3-030-80472-5_3.
[15] T. Lyudovyk, V. Pylypenko, Code-Switching Speech Recognition for Closely Related Languages, in: Proceedings of the Workshop on Spoken Language Technologies for Under-Resourced (2014) 1–6.
[16] N. B. Vasileva, et al., Corpus of Ukrainian On-Air Speech. Speech Technol. 2 (2012) 12–21.
[17] J. Meyer, JRMeyer/Open-Speech-Corpora, 2021. URL: https://github.com/JRMeyer/open-speech-corpora.
[18] MON Ukraine—YouTube, 2021. URL: https://www.youtube.com/channel/UCQR9sMWcZshAwYX-EYH0qiA.
[19] Deutsche Welle in Ukrainian—YouTube, 2021. URL: https://www.youtube.com/channel/UCQwVj4PyS5leCgEJY4I2t1Q.
[20] Toronto TV—YouTube, 2021. URL: https://www.youtube.com/channel/UCF_ZiWz2Vcq1o5u5i1TT3Kw.
[21] Common Voice by Mozilla, 2021. URL: https://commonvoice.mozilla.org/.
[22] TED Talks, 2021. URL: https://www.ted.com/talks.
[23] J. Devlin, et al., BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding (2019) 1–16. arxiv:181004805.
[24] Y. Liu, et al., RoBERTa: A Ro-bustly Optimized BERT Pretraining Approach (2019) 1–13. arxiv:190711692.
[25] D. Povey, et al. The Kaldi Speech Recognition Toolkit, in: Proceedings of the ASRU (2011) 1–4.
[26] A. Lee, T. Kawahara, K. Shikano, Julius—an Open Source Real-Time Large Vocabulary Recognition Engine, in: Proceedings of the Eurospeech (2001) 1–4.
[27] A. Hannun, et al., Deep Speech: Scaling up End-to-End Speech Recognition (2014) 1–12. arxiv:14125567.
[28] Watanabe, S.; et al., ESPnet: End-to-End Speech Processing Toolkit, in: Proceedings of the Interspeech (2018) 2207–2211.
[29] S. Schneider, et al., Wav2vec: Unsupervised Pre-Training for Speech Recognition, in: Proceedings of the Interspeech (2019) 3465–3469.
[30] S. Gnatyuk, et al., Modern method and software tool for guaranteed data deletion in advanced big data systems, Advances in Intelligent Systems and Computing (2019) 581–590. doi:10.1007/978-3-030-12082-5_53.
[31] L. Derczynski, Complementarity, F-Score, and NLP Evaluation, in: Proceedings of the 10th International Conference on Language Resources and Evaluation (2016) 261–266.
[32] D. Klakow, J. Peters, Testing the Correlation of Word Error Rate and Perplexity, Speech Commun. 38 (2002) 19–28.
[33] O. Iosifova, et al., Analysis of Automatic Speech Recognition Methods, in: Proceedings of the Workshop on Cybersecurity Providing in Information and Telecommunication Systems 2923 (2021) 252–257.
[34] T. Wolf, et al. HuggingFace's Transformers: State-of-the-Art Natural Language Processing (2020) 1–8. arxiv:191003771.
[35] A. Baevski, et al., Wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations 2020, 1–19. arxiv:200611477.