Richtlinie zum „Software-Sprint“

Zee – Wir bauen Werkzeuge für besser nutzbare Sprachanwendungen

Schlussbericht

Zuwendungsempfänger:

Rany Keddo

Das diesem Bericht zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 01IS23S43 gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt beim Autor.

Kurze Darstellung der Aufgabenstellung und Motivation

Zee bietet grundlegende Bibliotheken für Entwickler von Sprachsynthese- (TTS) und Spracherkennungs Modellen (SR): hochwertige Infrastruktur Bibliotheken für Alignment, bei dem Text Transkriptionen mit dem Sprachton zeitlich synchronisiert werden, und eine entsprechende Webanwendung für Untertitelung.

Ich bin in Neuseeland aufgewachsen und interessiere mich für Projekte zur Wiederbelebung der Maori-Sprache. Diese Projekte versuchen, eine technische Infrastruktur für und durch lokale Gemeinschaften aufzubauen, die nicht von großen Technologieunternehmen kontrolliert wird. Diese Bemühungen haben mich inspiriert und ich möchte Werkzeuge entwickeln, die solche Projekte unterstützen könnten.

Sprachtechnologie wird in der Regel von einer Handvoll großer Technologieunternehmen entwickelt, die proprietäre Werkzeuge verwenden. Unabhängige Projekte, die z.B. im Geiste der Tech Sovereignty ins Leben gerufen wurden, versuchen, Sprachtechnologien von vergleichbarer Qualität zu entwickeln, jedoch mit wesentlich geringeren Ressourcen - begrenzte finanzielle Mittel, kleinere Datensätze und weniger technische Ressourcen. Obwohl es Open-Source-Projekte wie Coqui gibt, sind einige grundlegende Bausteine für die Entwicklung dieser Sprachtechnologien in der Open-Source-Welt noch veraltet. Alignment und Phonemisierung sind solche vernachlässigten, aber wichtigen Bausteine für Sprachtechnologie. Die Verbesserung dieser Infrastruktur wird Grassroots Sprach Projekten dabei helfen, bessere Ergebnisse zu erzielen. Gleichzeitig wird Zee offene Untertitelung Tools für Hörgeschädigte bieten. Der Fokus in diesem Vorhaben lag auf dem Alignment Use Case.

Um diese Ziele umzusetzen, habe ich auf Basis meiner bestehenden “timething” Library eine Alignment Library entwickelt, mit der lange Inhalte, wie z.B. Podcasts Aligned werden können.

Daraufhin wurde eine Web-UI entwickelt, damit nicht-technische User:innen ihre Inhalte selber hochladen und Alignen können. Die wichtigsten Meilensteine waren:

- Untersuchung von Optionen für Wav2Vec2.0-Langform-Inferenz und Alignment
- Implementierung und Test von Alignment-Ansätzen für Langform-Inhalte
- Anwendung auf reale Daten wie einen Podcast und Behebung von Problemen
- Optimierung des Aligners, damit er auch bei langen Inhalten schnell genug läuft
- Erstellung einer SPA-Webanwendung und einer Python-Api für Upload und Alignment

Beitrag des Projektes zu den Zielen der Förderinitative „Software-Sprint“

Zee richtet sich an Entwickler von grassroots Sprachtechnologien und an Benutzer von vereinfachten Web Schnittstellen, die keine technischen Kenntnisse erfordern:

- Entwickler von grassroots Sprachtechnologien. Ich werde mich an bestehende Netzwerke wenden und mit Autoren von Open-Source-TTS- und SR-Projekten zusammenarbeiten, um Zee zu integrieren.
- Projekte, die an der Wiederbelebung indigener Sprachen arbeiten. Ich werde meine bestehenden Netzwerke in diesen Gemeinschaften nutzen.
- Hörgeschädigte
- Kinder mit Sprach Lernschwierigkeiten

Der Fokus in diesem Vorhaben lag auf dem Alignment, und somit auf den Entwicklern von grassroots Sprachtechnologien.

Diese Werkzeuge sind wesentliche Bestandteile vollständiger Sprachsysteme. Während große Technologieunternehmen über proprietäre interne Tools verfügen, verlassen sich Grassroots Entwickler auf die besten Open Source Optionen. Zee bietet eine Lösung, die diesen Entwicklern hilft, qualitativ hochwertige Systeme zu erstellen. Zee kann auch direkt von hörgeschädigten oder lernbehinderten Nutzern über eine benutzerfreundliche Website verwendet werden.

Ausführliche Darstellung der Ergebnisse

Während des Projekts wurden die folgenden Arbeiten abgeschlossen:

- Aktualisierung von Timething für die Bearbeitung von Inhalten in langer Form. Es ist nun möglich, Inhalte, die z.B. 1 Stunde lang sind, zu alignen.
- Timething ist über ein Command-Line-Interface verfügbar. Es hat jetzt 65 Sterne auf Github und Benutzer, die es für ihre Grassroots-Sprachprojekte verwenden.
- Timething Web wurde entwickelt, damit auch nicht-technische Benutzer Timething nutzen können. Die Web-UI besteht aus einem React-Frontend und einem Python-Backend und bietet die folgenden Funktionen:
    - Ein modernes Design und UX wurden in den Mittelpunkt dieser Web-UI gestellt
    - Robustes Hochladen großer Mediendateien. Es wurden eigene Protokolle entwickelt, um Datei-Uploads zu bündeln, was stundenlange Upload-Prozesse und die Wiederaufnahme unterbrochener Uploads ermöglicht. Uploads können per Drag &

Drop gestartet werden, und es werden hilfreiche Rückmeldungen zum Fortschritt gegeben.
    - Transkodierung von Medieninhalten über ffmpeg auf dem Server, einschließlich Echtzeit-Fortschrittsanzeigen in der Web-UI über SSE. Der Prozess läuft in der Cloud unter Verwendung von GPUs.
    - Transkription von Medieninhalten über Whisper, implementiert auf dem Server, einschließlich Echtzeit-Fortschrittsanzeigen in der Web-UI über SSE. Der Prozess läuft in der Cloud unter Verwendung von GPUs.
    - Diarisierung von Medieninhalten über Pyannote. Der Prozess läuft in der Cloud unter Verwendung von GPUs.
- Ein Video-Streaming-Protokoll, das auf dem Server und dem Client implementiert ist. Es ermöglicht die webbasierte Wiedergabe der hochgeladenen Mediendateien, einschließlich der Suche nach beliebigen Positionen in der Mediendatei.
- Ein Custom Medienplayer, einschließlich Zeitangaben und Wiedergabesteuerung.
- Eine Editor-Oberfläche zur Anzeige von Titel, Transkription und Sprecher:Innen. Titel und Sprecher:Innen können bearbeitet werden und die Änderungen werden im Backend gespeichert, sobald sie geschehen. Der Editor ist von Google Docs inspiriert, und alle Bearbeitungen werden wie in einem einfachen Texteditor vorgenommen.
- Eine Einstellungs-Seitenleiste, die die Bearbeitung von Titel, Beschreibung, Ausgangssprache und Sprechern ermöglicht.
- Prompting von Whisper mit der gegebenen Beschreibung. Wir folgen den Empfehlungen von openai und senden die Beschreibung an chatgpt, um ein fake Transkript zu erstellen. Dieses wird als Prompt für Whisper verwendet, um die Leistung zu verbessern.
- Änderung der Source-Sprache. Dies ist erforderlich, wenn Whisper die falsche Ausgangssprache erkannt hat. Die Änderung dieser Sprache löst eine erneute Verarbeitung aus, und der Fortschritt wird inline in der Seitenleiste der Einstellungen angezeigt.
- Text und Video werden über Timecodes synchronisiert. Ein Indikator hebt jedes Wort hervor, während es im Video gesprochen wird. Wenn im Editor auf ein beliebiges Wort geklickt wird, springt das Video an die entsprechende Stelle. Wenn der Video-Player von den Nutzerinnen vorgespult wird, wird der Indikator auf das richtige Wort im Editor verschoben.
- Download einer SRT-Datei mit Zeitangaben und der Transkription

Mit diesen Ergebnissen war ich in der Lage, alle Meilensteine, zu denen ich mich verpflichtet hatte, zu erreichen. Meine ursprünglichen Pläne, einen Phonemizer zu bauen, wurden nicht erreicht, aber in dieser PTF-Runde beschränkte sich mein Antrag auf den Alignment Use-Case, nicht auf den Phonemizer Use-Case.

Während der Entwicklung des Projekts begann ich, mit echten Nutzer:innen zu arbeiten, um die Entwicklung der Benutzeroberfläche zu steuern. Diese Benutzer waren alle Content-Creators, die Untertitel und Übersetzungen für von ihnen produzierte Video Inhalte benötigten. Es wurde klar, dass die Web-UI ein viel größeres Projekt war als erwartet. Dazu einige Einblicke:

- Content-Creators haben sehr große Videodateien auf ihren Laptops. Ohne zuverlässigen Upload dieser großen Dateien wird die gesamte App unbrauchbar. Daher habe ich viele Entwicklungszyklen mit dem Umsetzen der Upload-Funktionen verbracht und mit echten Dateien getestet, die von den Benutzern bearbeitet wurden. Damit dieser Aspekt wirklich nahtlos funktioniert, ist für große Dateien immer noch eine clientseitige Transkription über

ffmpeg in wasm erforderlich, da dies die Upload-Geschwindigkeiten bei Home-Internet-Anschlüssen erheblich reduzieren würde. Diese Funktion ist noch nicht umgesetzt.

- Die Ergebnisse der Transkription mit Whisper können viele Fehler enthalten. Whisper neigt zu Halluzinationen, und die Ergebnisse bei Sprachen abseits von Englisch können schlechter sein. Einer meiner Nutzer:innen lud ein Interview auf Englisch und Japanisch auf Timething-web hoch, und es waren viele Anpassungen erforderlich, um vernünftige Ergebnisse zu erzielen.
- Geschwindigkeit ist wichtig. Die Upload-, Transkriptions- und Transkodierungsgeschwindigkeiten tragen alle dazu bei, wie lange die Benutzer auf die Ergebnisse warten müssen. Die Nutzer sind nicht bereit, stundenlang auf Ergebnisse zu warten, vor allem wenn das Browserfenster während des gesamten Prozesses geöffnet bleiben muss.
- Die Ergebnisse von ML-Modellen wie Whisper enthalten fast immer Fehler. Für die Erstellung von Untertiteln müssen die Ergebnisse jedoch sehr genau sein, da die Zuschauer keine falschen Untertitel tolerieren. Folglich müssen die Nutzer:innen in der Lage sein, die Ergebnisse der Transkription und Diarisierung zu bearbeiten. Diese Funktionen sind an sich nicht trivial.
- Die Benutzer haben kollaborative Bearbeitungs Workflows. Alle Nutzer:innen, mit denen ich Tests durchgeführt habe, haben mit Dritten zusammengearbeitet. Dies beginnt bei der Genauigkeit der Transkription, bei der die Redakteurin möglicherweise nicht die Originalsprache eines Interviews spricht und eine Mitarbeiterin benötigt, um dies zu überprüfen. Als Nächstes können Übersetzungen Rückmeldungen und Korrekturen von Dritten erfordern, z. B. zur Anpassung an gegebene Text Richtlinien. Dies ist ein Hin und Her, das innerhalb der gleichen Plattform ablaufen sollte.
- Übersetzung ist für Nutzer:innen wichtig. Es sollte möglich sein, eine oder mehrere Übersetzungen für den Originalinhalt anzubieten.
- Die Sprechererkennung ist für die Nutzer:innen wichtig. Die Nutzer:innen, mit denen ich getestet habe, hielten das Produkt ohne Sprechererkennung für unbrauchbar.
- Das Editieren in einer solchen Umgebung scheint ein weitgehend ungelöstes Problem zu sein. Alle, mit denen ich gesprochen habe, hatten damit zu kämpfen, selbst unter Nutzung von bezahlten kommerziellen Tools.

Die Open Knowledge Foundation stand mir während des gesamten Projekts mit Rat und Tat zur Seite. Sie unterstützte mich dabei, mich auf die Nutzer:innen zu konzentrieren, die das System tatsächlich verwenden würden, und brachte mich mit Open-Source-Expert:innen zusammen, die mit dem Bereich vertraut sind.

Zielgruppe, Nutzen und mögliche Weiterentwicklungen

Bei der Open-Source-Bibliothek Timething Alignment sind die Nutzer:innen nach wie vor Grassroots-Sprachtechnologieprojekte. Im Projektverlauf konnte ich weitere Nutzer:innen dieser Kategorie finden. Dabei handelte es sich hauptsächlich um einzelne Entwickler:innen oder Studierende und Entwickler:innen, die mit Universitäten oder kleinen kommerziellen Projekten verbunden sind. Diese Nutzer:innen folgen timething auf Github, und es werden vermutlich auch weitere Nutzer:innen timething ohne Marketingmassnahmen finden. Sie können timething nutzen, um Sprachaufnahmen mit Transkripten in ihren eigenen Projekten abzugleichen. Eine qualitativ

hochwertige Alignment-Bibliothek ist für diese Nutzer:innen nützlich, da sie ein zentraler Bestandteil vieler Sprachprojekte ist.

Während der Entwicklung von Timething-Web wurde deutlich, dass sich das Produkt primär an medienschaffende Content-Creators richtet. Zu den Hauptnutzer:innen zählen Videoproduzent:innen, die für ihre Projekte zwar spezielle Tools benötigen, jedoch weder die Zeit noch die technischen Voraussetzungen haben, um sich mit der Installation von Python-Software oder der Nutzung von Command-Line-Tools und APIs auseinanderzusetzen. Sie bevorzugen eine Plattform, auf der sie ihre Inhalte einfach hochladen und Zugang zu unkomplizierten Transkriptions-, Übersetzungs- und Untertitelungstools haben. Diese Nutzergruppe erwartet eine nahtlose Content-Integration und eine intuitive Bearbeitung von Alignments.

Es zeigt sich, dass ein derartiges Tool den Anforderungen vieler potentiellen Nutzer gerecht wird – von jenen, die kommentierte Screen Recordings teilen und Meeting-Recordings distribuieren, über Podcast Creators bis hin zu Personen, die ihre Videos in einem accessible Format publizieren möchten, sowie Content Creators, die den Impact ihrer Arbeit auf mehrere Sprachen erweitern und ihre Inhalte für Hörgeschädigte oder Personen, die Inhalte im Silent Mode genießen wollen, leichter zugänglich machen möchten. Aktuelle Open-Source-Software für diese Tasks bietet keine moderne Web-Interface, die auf beliebigen Plattformen einsetzbar ist, das Publishing von Inhalten auf dem Web unterstützt, moderne Editing-Tools mit exzellenter UX oder Multi-User Editing-Workflows ermöglicht. Zudem basieren bestehende Tools nicht auf der neuesten AI-Generation.

Die Content Creators, mit denen ich zusammengearbeitet habe, setzen kommerzielle Tools wie Adobe Premiere oder webbasierte kommerzielle Lösungen wie trint.com ein. Der Markt bietet eine Vielfalt an kommerziellen Tools mit variierender Qualität. Es hat sich gezeigt, dass alle Creators, mit denen ich zusammenarbeitete, selbst mit diesen kommerziellen Tools Schwierigkeiten hatten. Obwohl sie ihre Projekte abschließen, sind die Editing-Flows umständlich, fehleranfällig und erfordern viel überflüssige manuelle Arbeit, um am Ende ein funktionierendes Ergebnis zu erzielen.

Ein Open-Source-Tool wie Timething-Web könnte einen signifikanten Impact leisten, da es präzise die Bedürfnisse aller erwähnten Zielgruppen abdeckt – als frei verfügbares und Open-Source-Tool. Es scheint machbar, die Qualität kommerzieller Tools für spezifische Use Cases wie Transkription, Übersetzung und Captioning zu übertreffen, besonders in einem Setting, das kollaborative Arbeit ermöglicht. Timething-Web zeigt bereits vielversprechende Ansätze, benötigt jedoch weitere 6 Monate Entwicklungsarbeit, um Editing-Flows zu optimieren. Zu den notwendigen Entwicklungszielen gehören:

- Implementierung von User Management
- Integration des TipTap-Editors und des Hocus Pocus-Servers für Collaborative Editing analog zu Google Docs
- Möglichkeit zur Bearbeitung von Speakern und Text im Editor UI
- Implementierung einer Editing-History mit Rollback-Option
- Synchronisierung des Backend-Zustandes mit den Modellen für Alignment, Diarization und Transcription
- Auslösung eines neuen Alignment-Runs bei Dokumentenänderungen, mit Client-seitiger Aktualisierung
- Einbindung eines Open-Source Translation Models

- Integration von Chat-GPT für Übersetzungs Zwecke
- Konfigurierbare Nutzung des Backend-Übersetzungsdienstes
- Side-by-Side Translation View im Studio
- Anpassung der UI für mobile Geräte
- Integration von Recorder.js für Direktaufnahmen von der Webseite, inklusive Screen Capture
- Editor View für die Bearbeitung von Captions
- Design und Implementierung eines UI-Elements zur Anpassung der Untertitel-Timings

Im Verlauf des Projekts lag der Schwerpunkt hauptsächlich auf der Entwicklung der Benutzeroberfläche. Ich entschied mich für React, um die Webseite zu entwickeln. Da ich kein Frontend-Entwickler bin, startete ich mit mäßigen Kenntnissen in React-Entwicklung. Während der Entwicklung lernte ich viel über React und ergänzte diese Lernkurve durch einen professionellen React-Trainingskurs.

Kurze Darstellung der Arbeiten, die zu keiner Lösung geführt haben

Ich begann das Projekt, indem ich untersuchte, wie man die Timething-Bibliothek so anpassen kann, dass sie für reale Inhalte funktioniert und nicht nur für 8-Sekunden-Segmente für Machine Learning Datasets. Ich verbrachte ziemlich viel Zeit damit, verschiedene Ansätze zu untersuchen und zu testen, bevor ich herausfand, dass Pytorch mittlerweile integrierte Alignment-Funktionen hat und das ein einfacher Ansatz für das Alignment von Langform-Inhalten einfach darin besteht, das Audio in Stücke zu schneiden und wav2vec 2.0 Inferenz auf jedem Stück laufen zu lassen, wobei die Ergebnisse in einem einzigen Logit-Tensor erfasst werden. Obwohl dieser Ansatz Probleme mit der Spracherkennung an den Schnittgrenzen ignoriert, scheint es immer noch gut genug zu sein, um gute Alignment-Ergebnisse zu erhalten. Infolgedessen habe ich die anderen Ansätze, an denen ich gearbeitet hatte, verworfen.

Kurze Angabe von Präsentationsmöglichkeiten für mögliche Nutzer

Die Arbeiten können auf Github gefunden werden:

- Timething, eine Bibliothek für das Alignment Audio Daten: http://github.com/feldberlin/timething
- Timething-web, eine Web-UI für die Transkription, Übersetzung und Untertitelung von Medieninhalten: http://github.com/feldberlin/timething-web

Die Web-Benutzeroberfläche ist derzeit auf http://modal.com angewiesen. In Zukunft wird diese Einschränkung gelockert werden. In der Zwischenzeit können Nutzer die Web-UI auf der kostenlosen Ebene von Modal einsetzen.

Kurze Erläuterung zur Einhaltung der Arbeits- und Kostenplanung

Mein Betreuer beim DLR hat mich in meinem Antrag davon überzeugt, das Projekt zu halbieren und mich nur auf einen Use Case, Alignment, zu konzentrieren. So konnte ich alle Milestones erreichen. Nach meiner ersten Einschätzung wäre es aber möglich gewesen, auch den zweiten Use Case, den Phonemisierer, umzusetzen. Bei der Umsetzung des Web UI für den Use Case Alignment wurde mir klar, dass ein solches Interface einen sehr großen Aufwand darstellt. Meine ursprüngliche Idee war es,

nur das Command Line Interface von Timething in Form einer Webseite zur Verfügung zu stellen. Für nicht-technische Benutzer:innen wäre dies aber nicht ausreichend gewesen, um brauchbare Captions zu erstellen. Außerdem wurde in der Zusammenarbeit mit Testbenutzern klar, dass auch Kollaborations-Features und Übersetzungen notwendig sind, um Captions für reale Projekte zu erstellen.

Daher musste ich meine Arbeitsplanung so anpassen, dass ich viel mehr Zeit in die Entwicklung der Web-Benutzeroberfläche mit React investierte und weniger Zeit in die Core Libraries und Algorithmen mit Pytorch.

Kurze Darstellung von etwaigen Ergebnissen bei anderen Stellen

Es gab keine Entwicklungen anderer Personen oder Institutionen, die Einfluss auf meine Arbeiten oder Zielsetzung hatten.