sound of ai accelerator

mentor self-presentation

alexander lerch

background
education

- **PhD, TU Berlin** 1994–2000
  - Software-Based Extraction of Objective Parameters from Music Performances
  - summa cum laude

- **Diplom-Ingenieur Electrical Engineering, TU Berlin** 2004–2008
  - concentration: telecommunications/signal processing, minors: communications and technical acoustics
  - summa cum laude

- (**Diplom-Tonmeister (Music Production), UdK Berlin**) 1996–2000
  - no degree

background
professional

- **Full-time**
  - *Assistant/Associate Professor, Georgia Institute of Technology* 2013–pres
  - *Co-Founder/Head of Research, zplane.development* 2000–2013

- **Temporary**
  - *Visiting Professor, Central Conservatory of Music, China* Summer 2023
  - *Visiting Professor, Shanghai Jiao Tong University, China* Summer 2019
  - *Visiting Professor, ShanghaiTech University, China* Summer 2018
  - *Post-Doc, University of Victoria, Canada* 2010

research
areas of interest

- **audio content analysis**
  - music/speech classification
  - musical instrument recognition
  - drum transcription
  - chord detection
  - auto tagging
  - ...
- **audio processing**
  - source separation
  - audio effects
  - ...
- **music performance analysis**
  - extraction of objective performance parameters
  - student assessment

research
areas of interest

- **audio content analysis**
  - music/speech classification
  - musical instrument recognition
  - drum transcription
  - chord detection
  - auto tagging
  - ...
- **audio processing**
  - source separation
  - audio effects
  - ...
- **music performance analysis**
  - extraction of objective performance parameters
  - student assessment

research
areas of interest

- **audio content analysis**
  - music/speech classification
  - musical instrument recognition
  - drum transcription
  - chord detection
  - auto tagging
  - ...
- **audio processing**
  - source separation
  - audio effects
  - ...
- **music performance analysis**
  - extraction of objective performance parameters
  - student assessment

methods
methods of interest

- **representation learning**
  - improved structure of embedded representations
  - transferring knowledge from other representations and tasks
  - enforcing the meaning of specific embedding dimensions
  - ...
- **insufficient data** for training
  - semi- and self-supervised learning
  - reprogramming
  - ...
- **objective system evaluation**
  - evaluation of controllable systems with correlated attributes
  - statistical models for comparison of properties
  - metrics for sound generation

scholar.google.com/citations?user=29dF3UIAAAAJ

methods
methods of interest

- **representation learning**
  - improved structure of embedded representations
  - transferring knowledge from other representations and tasks
  - enforcing the meaning of specific embedding dimensions
  - ...
- **insufficient data** for training
  - semi- and self-supervised learning
  - reprogramming
  - ...
- **objective system evaluation**
  - evaluation of controllable systems with correlated attributes
  - statistical models for comparison of properties
  - metrics for sound generation

scholar.google.com/citations?user=29dF3UIAAAAJ

methods
methods of interest

- **representation learning**
  - improved structure of embedded representations
  - transferring knowledge from other representations and tasks
  - enforcing the meaning of specific embedding dimensions
  - ...
- **insufficient data** for training
  - semi- and self-supervised learning
  - reprogramming
  - ...
- **objective system evaluation**
  - evaluation of controllable systems with correlated attributes
  - statistical models for comparison of properties
  - metrics for sound generation

zplane.development
overview

- founded in 2000
  - founders: Tim Flohrer, Martin Schwerdtfeger, Alexander Lerch
- high-quality (production quality) state-of-the-art algorithms for music analysis and processing
  - research collaboration with universities
- products
  - cross-platform SDKs (software developer kits)
  - some end-user products

www.zplane.de

zplane.development
products

- technology provider to the music industry
- best known for algorithms time and pitch modification of music: *time stretching*, *pitch shifting*

tempo manipulation	original		double tempo		mod. timing	
pitch manipulation	original		rand. modified		harmonized	
	original		minorized			

- technology licensed mainly for
  - Digital Audio Workstations (DAW)
  - DJ software
  - music production tools

zplane.development
historical background

- started with minimal resources
  - no start-up ecosystem/support
  - no investors, but a bank loan
- started with minimal business knowledge
  - three engineers
  - no insights into market beyond consumer view
  - no experience with marketing or customer relations
- started with limited software engineering knowledge
  - experience with implementation of algorithms
  - no experience with architectural design, maintenance, etc.
  - limited experience with performance optimization

zplane.development
business model

- iteration 1
  - service provider for music software industry
- iteration 2
  - develop SDK and exclusively sell it
- iteration 3
  - sell non-exclusively
  - negotiate for visibility
- iteration 4
  - royalties-based b2b model
- iteration 4.5
  - add simple end-user products for visibility and evening out cash-flow

zplane.development
business model

- iteration 1
  - service provider for music software industry
- iteration 2
  - develop SDK and exclusively sell it
- iteration 3
  - sell non-exclusively
  - negotiate for visibility
- iteration 4
  - royalties-based b2b model
- iteration 4.5
  - add simple end-user products for visibility and evening out cash-flow

zplane.development
business model

- iteration 1
  - service provider for music software industry
- iteration 2
  - develop SDK and exclusively sell it
- iteration 3
  - sell non-exclusively
  - negotiate for visibility
- iteration 4
  - royalties-based b2b model
- iteration 4.5
  - add simple end-user products for visibility and evening out cash-flow

zplane.development
business model

- iteration 1
  - service provider for music software industry
- iteration 2
  - develop SDK and exclusively sell it
- iteration 3
  - sell non-exclusively
  - negotiate for visibility
- iteration 4
  - royalties-based b2b model
- iteration 4.5
  - add simple end-user products for visibility and evening out cash-flow

zplane.development
business model

- iteration 1
  - service provider for music software industry
- iteration 2
  - develop SDK and exclusively sell it
- iteration 3
  - sell non-exclusively
  - negotiate for visibility
- iteration 4
  - royalties-based b2b model
- iteration 4.5
  - add simple end-user products for visibility and evening out cash-flow

zplane.development
take-aways

- do your thing
  - get all the advise you can get but make up your own mind
  - you decide but always be open to question decisions/processes
  - have a clearly defined value system (but get out of your comfort zone!)
  - know when to listen to customers and when not
- don't believe...
  - ... your competitors' marketing — everyone cooks with water
  - ... your own marketing — it's ok not being a genius
- transparency where possible can build trust
  - explain decisions and problems to users
  - don't be afraid to share some technical details here and there
- IF you have long-term plans
  - minimize 3rd party dependencies (API calls, platforms, libraries,...)
  - don't over-engineer but spend a significant amount of time on automated tests
  - define coding and commenting style and commit behavior
  - avoid concentrating critical knowledge in only one person without fallback strategy

zplane.development
take-aways

- do your thing
  - get all the advise you can get but make up your own mind
  - you decide but always be open to question decisions/processes
  - have a clearly defined value system (but get out of your comfort zone!)
  - know when to listen to customers and when not
- don't believe...
  - ... your competitors' marketing — everyone cooks with water
  - ... your own marketing — it's ok not being a genius
- transparency where possible can build trust
  - explain decisions and problems to users
  - don't be afraid to share some technical details here and there
- IF you have long-term plans
  - minimize 3rd party dependencies (API calls, platforms, libraries,...)
  - don't over-engineer but spend a significant amount of time on automated tests
  - define coding and commenting style and commit behavior
  - avoid concentrating critical knowledge in only one person without fallback strategy

zplane.development

take-aways

- do your thing
  - get all the advise you can get but make up your own mind
  - you decide but always be open to question decisions/processes
  - have a clearly defined value system (but get out of your comfort zone!)
  - know when to listen to customers and when not
- don't believe...
  - ... your competitors' marketing — everyone cooks with water
  - ... your own marketing — it's ok not being a genius
- transparency where possible can build trust
  - explain decisions and problems to users
  - don't be afraid to share some technical details here and there
- IF you have long-term plans
  - minimize 3rd party dependencies (API calls, platforms, libraries,...)
  - don't over-engineer but spend a significant amount of time on automated tests
  - define coding and commenting style and commit behavior
  - avoid concentrating critical knowledge in only one person without fallback strategy

zplane.development
take-aways

- do your thing
  - get all the advise you can get but make up your own mind
  - you decide but always be open to question decisions/processes
  - have a clearly defined value system (but get out of your comfort zone!)
  - know when to listen to customers and when not
- don't believe...
  - ... your competitors' marketing — everyone cooks with water
  - ... your own marketing — it's ok not being a genius
- transparency where possible can build trust
  - explain decisions and problems to users
  - don't be afraid to share some technical details here and there
- IF you have long-term plans
  - minimize 3rd party dependencies (API calls, platforms, libraries,...)
  - don't over-engineer but spend a significant amount of time on automated tests
  - define coding and commenting style and commit behavior
  - avoid concentrating critical knowledge in only one person without fallback strategy

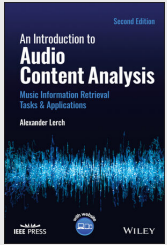
thank you!

contact

Alexander Lerch:
alexander.lerch@gatech.edu

www.AudioContentAnalysis.org
www.alexanderlerch.com

Music Informatics Group
musicinformatics.gatech.edu



github.com/alexanderlerch