University of Tartu
Chair of Data Science
Jaan Aru, Taavi Luik

2. December 2022
Benjamin Eckhardt (C28048)
benjamin.eckhardt@ut.ee 50006150108

Artificial and Natural Intelligence LTAT.02.024

COURSE PROJECT - CHECKPOINT II

LITERATURE SURVEY

NEURO SYMBOLIC AI AND MUSIC

STORY AND MOTIVATION

In the first checkpoint I already hinted that I have a big interest in understanding music to facilitate computer aid in composing. This stems from the facts that music is highly beautiful, apparently highly structured and that I have so far failed to find people or resources which can provide comprehensive understanding of it. Though whenever I attend[1] to truly skillful music, both in composition and performance, I quickly resign from the idea that automatically reproducing such is even desirable. I rather want to find systems that empower human composing, by exposing the underlying structure of music and enabling powerful manipulation in higher terms – leaving the guidance of the process and originality for the human and therewith joy and expression. Another domain I think of as fruitful is music transformation: foremost transcribing and arranging music for different ensembles, and some other kinds of style transfer could also be fun.

Music, but many more topics I am interested in have this aspiration in common – to enable understanding and meaningful manipulation of underlying structures in all kinds of things, in order to empower human creative agency, which is way more worthwhile than sampling. Therefore I will explore neuro symbolic artificial intelligence, as the direction to answer this, taking music as a niche flavor.

Since my group mates dropped out quietly, I estimated my chances to do something really nice relatively low. Instead of running ahead with my limited knowledge, I decided it would be the most fruitful in the longer term to first firmly ground my self in preexisting work and contemporary research (thank Taavi for this suggestion which I first found unacceptable).

The literature I have read so far is presented in the order I read it, listing the reason I read it for, a short summary of what I get as the core ideas, things I learned from it with a short explanation aimed at my future self, and my reflections on the thing. Due to time constraints I left out the papers on EA/QD that I studied around checkpoint 1. A preliminary conclusion is drawn at the end.

During the process I learned to use Zotero, Arxiv, Google Scholar, Semantic Scholar and the internet in general to do, organize and cite research. I developed adaptive paper reading skills. These pages are very sketchy, being primary a vehicle to record and facilitate my thought and learning processes. I still have to learn to do this properly for my self. The final submission I plan to definitely do in a much more digestible format, please excuse if this is a pain to read.

Learning Compositional Rules via Neural Program Synthesis [2]

read: very detailed — **reason:** After a long browsing in recent NIPS, this seemed both simple and novel enough. **see** my presentation for Computational Semantics for details …

Prior domain knowledge and human smartness: Formulate a DSL for sequence translation tasks. *Neural guessing:* Let a neural model attend to few input/output example pairs to propose programs in that DSL that could do the translation (a representation grammar). *Symbolic verification and execution:* Parse and execute the programs classically - if they fail to correctly translate the examples: resample from the model. During test very few, expressive

thoughts: The selected datasets are generated by context free grammars in the first place and I highly doubt this approach is applicable to more interesting domains. Because
1. The search is dumb: The neural component is just sampled upon *thousands* of times *without any information flowing regarding the reason of failure of the previous proposals.*
2. and inefficient: every proposal is again tested against every example pair (though there can be relatively few ones, if they are chosen right (here based on non-domain-specific heuristics)).
3. All "compositional meaning" is only *syntactic* ("YhoundredandX" is replaced by "100*Y+X", "two X" by "X X"). And there is no way to use this system as a backbone for anything that goes beyond only term rewriting?

→ The neural net shall propose not rigid grammars but an ambiguous one, that gets continually refined with each example pair (crossing out productions with counter examples), choosing maybe the simplest remaining option in the end. Since a general ambiguous grammar can be calculated from the meta grammar (the DSL), one would not even need a neural net? Maybe do AlphaGo style: overall tree structure of possible grammars is managed symbolically, with neural nets providing intuition, which subtrees shall be eliminated first. [Can this be a Thesis?] Instead of iteratively pruning a structure of all possibilities, one could iteratively add to a minimal program. A smarter example selection system could automate the selection of a minimal complete example set, or even synthesize a single complete example form, or pick next counter (or constructive) examples for an iterative refinement.

I wonder if this would even work for source code com- or transpilation by example in reasonable time, since that complexity seems already a lot higher than the provided examples. With optimization or other not completely trivial transformations the time complexity would explode..

It seems that mutually exclusive precise grammar rules are as well way too limited in more complex domains like music. There one would need ways to overlay or combine patters in a way that is soft enough to preserve eachs information.

If the searches dumbness and grammars restrictedness are resolved, maybe with my proposals, come back and bring musical grammars ([3] and referees).

Symbols and mental programs: a hypothesis about human singularity [4]

read: most, a while ago... — **reason:** was mentioned in the lecture and seemed perfectly fitting

Humans posses the unique capability to form sign systems / symbolic languages and put them into the world, but for a lot of different things, which are wired at *different* brain locations

Many nice definitions! Refers to Minimum description length (MDL) and Kolmogorov complexity (see below).

MDL linearly correlates with the ability of humans to predict sequence continuations
CNNs correlate way more with Baboon brains than a symbolic model (how do they correlate it??)
With humans it depends maybe on formal education, achieving sole correlation with symbolic models. I think they compare the performances with respect to the (symbolic) "complexity" of the task and see how that correlates across species?

thoughts: maybe signs are not soo fundamentally different and fancy, after all of course human artifacts are more resonating with humans. Maybe a triangle is not fundamental at all, but just appears so to the human brains because of their particular structure. Humans look at triangles at the walls and see divine human singularity, while other beings don't take notice of it. Maybe every species brain has its more or less intense quirks and humans just fail to notice the expression of such specific thought patterns in the same way. How can you know that the ape isnt trying to make you do something as stupid as the other way round, you performing as bad, because you just don't (maybe cant) get it (or because the apes are missing such strange intense need to make other beings do stupid discomforting tasks). The complexity of the tasks used for correlating apes, humans, CNNs and symbolic models is measured in a language that humans made up (math and geometry), so it only shows that there is a distinct propensity to model things in a specific way which they call symbolic, but that need not necessarily be the only or best way to think
Is the number of sign processing brain regions limited, or does it grow dynamically with the tasks?

First Steps of an Approach to the ARC Challenge based on Descriptive Grid Models and the Minimum Description Length Principle [5]

read: all but maths — **reason:** suggested by Taavi as an approach to the Abstraction and Reasoning Corpus, I think in reply to my *Learning Compositional Rules via Neural Program Synthesis*
Have a DSL to represent an example pair as data; evolve that one to represent it classically analytic in minimal description length. Evolving is done with classical search. Purely symbolic, no neural component?

The minimal description of the output examples in terms of the minimal description of the input examples is a template for the solution. ~7% accuracy or so

thoughts: There is no referring to other parts of the same structure (would need to filter out infinite recursion), which could further reduce description length. It seems not too unimplementable with their already implemented "context", which enables output nodes to refer to subtrees in the input. Instead the DSL could also integrate definitions of functions: like patterns that can be evolved the same way as data, and instantiated multiple times with varying parameters: eg to encode two rectangles of the same attributes but different position as one concept with two instances. This seems so obvious, why don't they have it?
They use classical search algorithms, why not EA/QD? A neural component could guide search, at least proposing data structures as in *Learning Compositional Rules via Neural Program Synthesis*, but also being smarter as imagined under ibd.

Such learnable patterns I see as analogous to the proposable grammar rules in *Learning Compositional Rules via Neural Program Synthesis.* Further thoughts are therefore there to find.

Minimum description length revisited [6]

read: failed — **reason:** above paper used it as a driving factor

Same as with *Spectral Norm Regularization…* below, I postponed a deeper understanding. Apparently a computationally tractable measure paralleling Kolmogorov complexity (mentioned below under NAR), implementing Occam's razor and thus the ibd. suggested favorization of shorter programs as better generalizing ones.

Neural Abstract Reasoner [7]

read: most — **reason:** suggested by Taavi as an approach to the Abstraction and Reasoning Corpus

A deterministic AE learns to encode the ARC grids. A DNC is trained to produce "instruction sets" (I guess just some more latent tokens) that a transformer should use to transform the examples and a novel input to the needed output.

The transformer cross attends to the example pairs with the query input and to the DNCs code, thereby "executing it as a program" (that is therefor parametrized over the query input(!) and the examples).

thoughts: The AE that encodes the grid seems strange to me: Right in the beginning we replace the exact symbolic input by an approximation? Maybe AlphaGo style is cool here: do symbolically discrete tree search on a set of proposed transformations, where the transformations are learned symbolically and discrete to be disentangled or to minimize a description length? Though in the end does AlphaGo operate on a latent representation of the grid as well, which can hardly be a precise representation of the factual situation in the first place? (If you don't use loots of neurons)

The Transformer cross attends to the DNCs code *and* again the examples. Is that only to mitigate for the limited information bandwidth in the DNCs code?

The code outputted by the DNC could, by the open ended recursiveness of DNC, symbolize arbitrary complex nested transformations. But the recursion depth is limited to the sequence length the transformer can attend to and the number of its layers.

Further the system seems unable to propose new transformations during testing, which are not capturable by the previously learned (how ever this would work anyway) - it is just luck if the training examples are complete and the network finds all axiomatic relations.

Why is the AE deterministic and not variational? Is that because they assume that the distribution of grids seen during training is complete? It provides the benefit of a sharper representation since the sampling makes VAEs blury.

for DNC see [8].

cross attention: Query comes from one sequence, Key and Value from the other. (Sometimes also QK/V). self attention is thus just regular attention with QKV form the same sequence.

Thoughts on Kolmogorov complexity, MDL and spectral norm regularization are collected under *Minimum Description Length Revisited*

Kolmogorov complexity: The length of the minimal touring machine program to produce a datum?

key takeaway: In the human brain, the symbols arise from the neurons and their embedding is in terms of them, their discrete written form a secondary product. Here its the same. So I tend to think of discrete symbols less then before as imperatively discrete inside the system. Though discretization seems to help generalizability and human analytical thought so much, where pure neural models fail. Maybe it just boils down to constraints in information bandwidth propagating from language into the brain. And the matter to build build systems that understand thereby arising human thought products boils down to force them to shape their brains like ours in this regard. Symbolic thought being an inductive bias on the brains working that doesn't so easily emerge but takes some design effort from evolution / AI persons. I tend to see it now as a matter of overall structure of a neural net rather than taming them with classical algorithms (though they can be an effective way to encode prior domain knowledge) (though the nets shall write their own classical programs for analytical tasks, like humans do.)

Hybrid computing using a neural network with dynamic external memory [8]

read: only overview, because I got distracted by scanning following works. It is difficult to decide whether to read something or not, based on its novelty or fundamentality, then in the end I spend time almost only reading abstracts.. — **reason:** for understanding *Neural Abstract Reasoner*
A recurrent neural network produces read and write attention vectors to continously save and alter the rows of an arbitrary large memory matrix. Discretized are only the locations in the matrix (are these symbols?). Complex structures are representable associatively: by using the information from the read value for a new read key, which reads specific

What is the history of accesses saved for?

There appear to be lots of improvements upon it, which is the worthiest reading??

As applied in *Neural Abstract Reasoner* this component can manage and externalize knowledge, rather than combining it with the reasoning into one system of weights.

thoughts: Can the DNC sufficiently dynamic come up with programs? The processing of associative structures in memory is akin to evaluating a program, maybe a second memory (working memory) is useful, where the evaluation of the program can quickly externalize intermediate computational values to, to enable explicit recursive processing during test time that goes beyond the capabilities of the prelearned weights in net and memory?

Spectral Norm Regularization for Improving the Generalizability of Deep Learning [9]

read: no — **reason:** Neural Abstract Reasoner uses it as a main performing ingredient.

On this one I stopped early, saving the maths for a later date, when I feel more confident that I want to dig into it. Judging by the results and interpretations in *Neural Abstract Reasoner* it seems that this could be the structural inductive bias I was looking for in the key takeaway section for NAR. Though since it improves generalizability in general, I would assume it is rather analogous to an

inductive bias in all brains and not necessarily specific for symbolic thought. To prove this hypothesis further theoretical insights and experiments are needed, therefore I wanna continue on this. See also *Minimum Description Length Revisited*

Neural Discrete Representation Learning [10]

read: most, with video assistance — **reason:** earlier suggestion by Taavi; was missing discreteness Like a VAE, but quantized latents: every forward pass the encoded latents are replaced with nearest matching latents from a finite collection (=code book), which is optimized as the parameters are. And the prior is learnt, not static.

prior: the distribution of latents (from which you can sample new data). *How is it analogous to the Bayes intuition (in the appendix)?* Sampling from this latent prior is much more efficient than in pixel space.

Quantization: A finite collection of feature vectors shall replicate an inductive bias to use discrete symbols? The few possibilities for latents enforce their individual expressivity. Generalizability

Every spatial coordinate (h*w) of the encoders feature vectors (h*w*d) is replaced with the most similar (by L2-norm) code book vector: 1. replace with index (a model can learn with those as regular tokens), 2. one hot encode and multiply with code book; and fed into the decoder. In the backwards pass the gradient has to be split because argmin is not differentiable: gradients (which are computed for the code book vectors) get copied from decoder input to encoder output. Which moves the continous encodings in the directions the codes should have moved, eventually snapping to another code later. The code book vectors are optimized with the same gradient**(?)** The loss is the sum of 1. log likelihood of the input given the encoded and reconstructed image (this boils down to pixelwise MeanSquaredError (MSE) because likelihood is assumed to be gaussian (simple calculus); 2. $\|\cdot\|_2^2$ of the difference between continuous encoding and nearest discrete code (whereby gradient is only passed into code book); 3. the same, but gradient only through encoder output.

thoughts: Does each code book vector indeed capture a specific concept (thus being symbolic), or just forms and textures? Is generalization improved? -The few symbolic codes should be easier to compose into new meanings, if the are indeed more fundamental.
The codes cant be used very good to combine into *novel meanings* (rather than just novel assemblies) since they have a fixed low spatial density: Rather than each latent pixel being possibly a sequence of code vectors, each encoded into a more expressive combined vector, that the decoder shall understand. More flexible I imagine it would be to further decouple semantic from spatial information, by having the encoder maybe produce an attention map for every code book vector over a canvas. But how to combine those then?

research: VQ should be applicable the same to other latent architectures like transformers?

I felt, I needed to understand Bayes eventually which I tried in the Appendix.

Generating Diverse High-Fidelity Images with VQ-VAE-2 [11]

read: the new parts compared to VQ-VAE; mentioned in the video too - **reason:** suggested by Taavi

Mutli-level code book. More fancy learned priors improve reconstruction and generation.

compares: GANs are more compute intensive to evaluate and suffer from mode collaps(?) and lack of diversity, but provide sharper samples. Because he reconstruction loss makes the demand to capture the full sample distribution explicit.

thoughts: The multi level code book is really interesting: capturing the proposed recursivenes of abstraction, though its depth is hard coded: the system cant choose to abstract with further or less layers once its build and trained with a specific number. Also the layers are strictly hierarchic, whereas one could imagine a code book that is a dynamic general graph than a static tree: multiple different sources of information coming together based on the thing and merging downstream. Though that seems to some degree implicit by the processing of the decoder, which can assign composite meanings to patterns of code features?

Symbolic Music Loop Generation with Neural Discrete Representations [12]

read: high-level descriptions — **reason:** something with music which cited VQ-VAE

From MIDI (drum+bass); let a loop extractor model figure out repeated sequences, trained on loop-labeled natural and generated MIDI, without heuristics (didn't get how that works); a sequence that is looped is understood as a smallest complete meaningful musical phrase I think; let a VQ-VAE learn to encode this fragments; let an *autoregressive model* (4xLSTM) learn to predict sequences in the index space of the model; generate new loopable music by generating seqs with that LSTM and decoding them with the VQ-VAEs code book and decoder

more: they propose several metrics for the reconstructed/generated music; they test multiple sampling methods which affect the metrics (including diversity) – maybe come back to this

thoughts: would a transformer instead of the LSTM improve fidelity further?
I stopped to understand the loop extractor, because I am interested in understanding all kinds of musical structure in the same general terms and this seems to be quite specific for loopable sequences. But maybe coming back to it is good.

compares: better than CNN-VAE, Music Transformer and MuseGAN according to their metrics

AudioCLIP: Extending CLIP to Image, Text and Audio

read: no, but watched a video — **reason:** was the only thing from the person doing the VQ-VAE video related to music I found.

A third encoder head for *sound* (ES*ResNeXt* takes a mel spectrogram); three cross similarity matrices
Lost interest for the moment because of the cross modality and the focus on picturable concepts and thus sound rather than thee divine unrepresentative music. But seems nice for cross modal art!

CLIP: a matrix of cosine similarities (boils down to dot products) between two token sequences produced by separate encoders from different modalities is optimized contrastively. 0Shot prediction of captions by taking the sentence formed by a template and a word drawn from some set whichs embedding is the most similar to the image. (Seems compute intense to try all.) This resembles a fully connected classification head with dynamic arbitrary many classes (the weights being made of the specific embeddings)

Mel spectrogram: log normalized spectrogram (not fft but something more general)

0shot prediction: predict unseen samples by combining more fundamental information

Towards Data-and Knowledge-Driven Artificial Intelligence: A Survey on Neuro-Symbolic Computing [13]

read: skimmed — **reason:** survey from 2020
Very introductory, but very nice *list of models* and lots of references!

Read section 6 – open challenges

Omnizart: A General Toolbox for Automatic Music Transcription [14]

read: complete — **reason:** found the library some time ago
It is a brief python compilation of several previous works for different ai music information retrieval techniques → useful branching point
see also: librosa?

Future Readings

DiffRoll: Diffusion-based Generative Music Transcription with Unsupervised Pretraining Capability

A Survey on Artificial Intelligence for Music Generation: Agents, Domains and Perspectives

ComMU: Dataset for Combinatorial Music Generation
very high quality dataset, very nice semantically conditioned samples

Code Generation Using Machine Learning: A Systematic Review

Harmony Transformer: Incorporating Chord Segmentation into Harmony Recognition [15]state of the art in Omnizart; improvement [16]

The Identity Thesis for Language and Music [3]and referees
useful musicological groundwork

and all recent in

- ISMIR transactions
- paperswithcode.com / music, neuro symbolic ...
- NeurIPS

which are too numerous to name

Conclusion So Far

Neuro symbolic systems are often implemented as a hand crafted symbolic algorithm controlling a neural network, which works fine for constrained and well known domains. But where the task for the system is to infer any kind of underlying structure, that is not known a priory (more than originating from symbol dictated human thought) one needs: Symbols to emerge from the neurons just as in the human brain. This inductive bias seems to me right now to have to be implemented in terms of clever network structure: shaping information flows in ways that mathematically maximize generalizability (and embracing recursion). As humans do, tedious computations shall be outsourced by a neural system by writing programs and own subroutines.

While I'm no symbol supremacist yet, I appreciate them as a vehicle to approach human cognition.

References

[1] A. Vaswani *et al.*, "Attention Is All You Need," 2017, doi: 10/gpnmtv.
[2] M. Nye, A. Solar-Lezama, J. Tenenbaum, and B. M. Lake, "Learning Compositional Rules via Neural Program Synthesis," in *Advances in Neural Information Processing Systems*, 2020, vol. 33, pp. 10832–10842. doi: 10.48550/arXiv.2003.05562.
[3] J. Katz and D. Pesetsky, "The Identity Thesis for Language and Music." LingBuzz, Jan. 2011. Accessed: Dec. 02, 2022. [Online]. Available: https://ling.auf.net/lingbuzz/000959
[4] S. Dehaene, F. Al Roumi, Y. Lakretz, S. Planton, and M. Sablé-Meyer, "Symbols and mental programs: a hypothesis about human singularity," *Trends Cogn. Sci.*, vol. 26, no. 9, pp. 751–766, Sep. 2022, doi: 10/gq9kx9.
[5] S. Ferré, "First Steps of an Approach to the ARC Challenge based on Descriptive Grid Models and the Minimum Description Length Principle," 2021, doi: 10/grbxpn.
[6] P. Grünwald and T. Roos, "Minimum description length revisited," *Int. J. Math. Ind.*, vol. 11, no. 01, p. 1930001, Dec. 2019, doi: 10/gqnr7w.
[7] V. Kolev, B. Georgiev, and S. Penkov, "Neural Abstract Reasoner," 2020, doi: 10/grbx38.
[8] A. Graves *et al.*, "Hybrid computing using a neural network with dynamic external memory," *Nature*, vol. 538, no. 7626, pp. 471–476, Oct. 2016, doi: 10/gc3f8k.
[9] Y. Yoshida and T. Miyato, "Spectral Norm Regularization for Improving the Generalizability of Deep Learning," 2017, doi: 10/grb29f.

[10] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, “Neural Discrete Representation Learning,” in *Advances in Neural Information Processing Systems*, 2017, vol. 30. doi: 10/grcf53.
[11] A. Razavi, A. van den Oord, and O. Vinyals, “Generating Diverse High-Fidelity Images with VQ-VAE-2,” in *Advances in Neural Information Processing Systems*, 2019, vol. 32. doi: 10/grcf5f.
[12] S. Han, H. Ihm, M. Lee, and W. Lim, “Symbolic Music Loop Generation with Neural Discrete Representations,” 2022, doi: 10.48550/ARXIV.2208.05605.
[13] W. Wang and Y. Yang, “Towards Data-and Knowledge-Driven Artificial Intelligence: A Survey on Neuro-Symbolic Computing,” 2022, doi: 10/gq85p8.
[14] Y.-T. Wu *et al.*, “Omnizart: A General Toolbox for Automatic Music Transcription,” *J. Open Source Softw.*, vol. 6, no. 68, p. 3391, Dec. 2021, doi: 10/grchvx.
[15] T.-P. Chen and L. Su, “Harmony Transformer: Incorporating Chord Segmentation into Harmony Recognition,” in *Proceedings of the 20th International Society for Music Information Retrieval Conference*, Delft, The Netherlands, Nov. 04, 2019, pp. 259–267. doi: 10.5281/zenodo.3527794.
[16] T.-P. Chen and L. Su, “Attend to Chords: Improving Harmonic Analysis of Symbolic Music Using Transformer-Based Models,” *Trans. Int. Soc. Music Inf. Retr.*, vol. 4, no. 1, Art. no. 1, Feb. 2021, doi: 10/grct22.

APPENDIX

Bayes

from blog: https://towardsdatascience.com/understand-bayes-rule-likelihood-prior-and-posterior-34eae0f378c5
likelihood: the probability of observing the data that has been observed assuming that the data came from a specific scenario. P(positive test | corona) how prob a positive test is, given one has corona
prior: the general prob of having corona independent of any test results (prob of corona in general)
evidence / marginal likelihood: the general prob of a positive test independent of having corona
posterior: not just the likelihood of a result given what has been observed in prior, but also considering how probable the prior is and the prob to have the given evidence in general

posterior = likelihood × prior / evidence

P(corona | positive test) = P(positive test | corona) * P(corona) / P(positive test)

In VQ-VAE: corona=data, positive test=, ; prior= the distribution of data (in latent space) ...