

Turkish music generation using deep learning

Item Type	Conference contribution
Authors	Aydingün, Anıl;Bağdatlıoğlu, Denizcan;Canbaz, Burak;Kökbiyık, Abdullah;Yavuz, M Furkan;Bölücü, Necva;Can, Burcu
Citation	Aydingün, A., Bağdatlıoğlu, D., Canbaz, B., Kökbiyık, A., Yavuz, M.F., Bölücü, N. and Can, B. (in press) Turkish music generation using deep learning, 28th IEEE 13th Signal Processing and Communications Applications Conference (SIU), 5th-7th October, 2020, Online.
DOI	10.1109/SIU49456.2020.9302283
Publisher	IEEE
Download date	2026-05-20 16:24:41
License	https://creativecommons.org/licenses/by-nc-nd/4.0/
Link to Item	http://hdl.handle.net/2436/623744

Derin Öğrenme ile Türkçe Şarkı Besteleme

Turkish Music Generation using Deep Learning

Anıl Aydingün, Denizcan Bağdatlıoğlu, Burak Canbaz, Abdullah Kökbiyık, M. Furkan Yavuz

Necva Bölücü⁶, Burcu Can

Bilgisayar Mühendisliği Bölümü

HUNLP Araştırma Laboratuvarı

Hacettepe Üniversitesi, Ankara, Türkiye

anilaydingun@gmail.com, dcb.denizcan@gmail.com, brkcnbz95@gmail.com, akokbiyık@gmail.com

b21427492@cs.hacettepe.edu.tr, necva@cs.hacettepe.edu.tr, burcucan@cs.hacettepe.edu.tr

Özetçe—Bu çalışmada derin öğrenme ile Türkçe şarkı besteleme üzerine yeni bir model tanıtılmaktadır. Şarkı sözlerinin Tekrarlı Sinir Ağları kullanan bir dil modeliyle otomatik olarak oluşturulduğu, melodiyi meydana getiren notaların da benzer şekilde nöral dil modeliyle oluşturulduğu ve sözler ile melodinin bütünleştirilerek şarkı sentezlemenin gerçekleştirildiği bu çalışma Türkçe şarkı besteleme için yapılan ilk çalışmadır.

Anahtar Kelimeler—müzik besteleme, dil modeli, derin öğrenme, tekrarlı sinir ağları.

Abstract—In this work, a new model is introduced for Turkish song generation using deep learning. It will be the first work on Turkish song generation that makes use of Recurrent Neural Networks to generate the lyrics automatically along with a language model, where the melody is also generated by a neural language model analogously, and then the singing synthesis is performed by combining the lyrics with the melody. It will be the first work on Turkish song generation.

Keywords—music composition, language model, deep learning, recurrent neural networks.

I. GİRİŞ

Son yıllarda yapay zekanın ulaşabildiği yeni alanlardan biri de otomatik müzik üretme olmuştur. 2016 yılında yayınlanan ilk yapay zeka müzik albümü AIVA¹ (Artificial Intelligence Virtual Artist) ile bu alandaki çalışmalar da hız kazanmıştır. Bu çalışma ilk olarak Mozart, Beethoven, Bach gibi sanatçıların besteleriyle eğitilerek klasik müzik üzerine uzmanlaştırılmış ve hatta sonrasında senfoni orkestrası tarafından da seslendirilen ilk yapay zeka müziği olarak tarihe geçmiştir.

Bu alandaki çalışmalar, insan tarafından üretilen müziğin yerine yapay zekayı geçirmekten ziyade insan ve yapay zekanın işbirliğini artırma amacını taşımaktadır. Diğer yandan, otomatik olarak müzik üretilmesinin birçok farklı alana da hizmet etmesi beklenmektedir. Örneğin, film, oyun, ya da reklam müziklerinin otomatik üretilmesi veya herhangi bir eğlence içeriğine ait bir müziğin üretilmesi olası uygulama alanları olabilecek niteliktedir.

Literatürde yapılan çalışmalar daha çok müzik besteleme üzerine olup, şarkı sözleriyle müziğin birleştirilmesi üzerine yapılan herhangi bir çalışmayla karşılaşmamıştır. Akademik literatür haricinde ise henüz bu sene Eurovision şarkı sözleri

ve besteleriyle eğitilmiş bir model tanıtılmıştır². Bu çalışmada, sadece müziğin bestelenmesi değil, aynı zamanda şarkı sözlerinin de otomatik olarak üretilmesi ve sözlerin müzikle birleştirilerek şarkı sentezleme yapılması amaçlanmıştır. Çalışma, Türkçe şarkı sentezleme olarak özelleştirilerek, sadece Türkçe sözlerin oluşturulmasına yönelik tasarlanmıştır. Müzik olarak ise Türk Halk Müziğine ait eserler kullanılarak geliştirilen model eğitilmiştir. Ancak model, herhangi bir başka müzik türü için de eğitilebilecek niteliktedir ve melodi ve müzik türü bağımlılığı içermemektedir.

Bildiğimiz kadarıyla Türkçe için şimdiye kadar şarkı besteleme üzerine çalışılmamıştır. Bu nedenle, tanıtılan model bu alandaki bilinen ilk çalışma olacaktır.

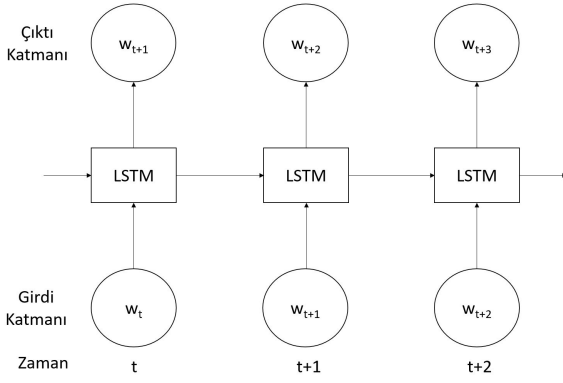
II. İLGİLİ ÇALIŞMALAR

Müzik besteleme çalışmaları son yıllarda derin öğrenme ile hız kazanmış olsa da, bu çalışmalar çok daha eskiye dayanmaktadır. Bu alanda ilk hesaplamalı model kullanan çalışma 1979 yılında Hiller ve Isaacson [1] tarafından gerçekleştirilmiştir. Bu ilk çalışmada algoritmik olarak besteleme yapabilmek için Markov zincirleri kullanılmıştır. Sonrasında farklı yapay zeka modelleri kullanılarak müzik besteleme üzerine oldukça fazla çalışma yapılmıştır. Bu yöntemler arasında, matematiksel modeller [2], [3], bilgi tabanlı sistemler [4], gramer tabanlı yöntemler [5]–[7], evrimsel modelleme yöntemleri [8]–[10], karma sistemler gibi farklı öğrenme yöntemleri bulunmaktadır. Bu alandaki çalışmaların büyük bir kısmını yapay sinir ağlarıyla gerçekleştirilen modeller oluşturmaktadır. Hatta bu çalışmalar 1989 yılında ilk defa Jordan tekrarlı sinir ağlarını kullanan bir çalışmaya [11] dayanmaktadır. Sonrasında yapay sinir ağları, Blues [12], Jazz müzik türü [13], Bach besteleri [14] ve Bartok besteleri [15] üretmek için farklı çalışmalar tarafından kullanılmıştır. Bildiğimiz kadarıyla Türkçe müzik besteleme için herhangi bir çalışma literatürde bulunmamaktadır.

Literatürde şarkı sözü üretmek üzerine de farklı çalışmalar yürütülmüştür. Potash ve arkadaşları [16] çalışmalarında sözcük tabanlı LSTM (Long Short Term Memory) kullanarak Rap şarkı sözü üretmişlerdir. Sharma ve Thuwal [17] karakter tabanlı LSTM modeli önermişlerdir. Fernandez ve arkadaşları [18] çalışmalarında karakter tabanlı GRU, RNN ve LSTM modelleri önermişlerdir. Sonuçları kıyasladıklarında önerilen

¹<https://www.aiva.ai/>

²https://www.youtube.com/watch?v=4MKAf6YX_7M



Şekil 1: LSTM tabanlı şarkı sözü üretme modeli. w_t, t anında LSTM tarafından işlenen sözcüğü ifade etmektedir.

GRU modelinin diğer modellere kıyasla daha başarılı olduğunu gözlemişlerdir.

III. NÖRAL DİL MODELİ İLE ŞARKI SÖZÜ ÜRETME

Bu çalışmada, n-gram tabanlı bir nöral dil modeli oluşturularak şarkı sözü üretilmektedir. Bu amaçla hem önceki sözcükleri hatırlamak, hem de kaybolan gradyan problemini yaşamamak için Uzun Kısa Dönemli Bellek Ağları (Long Short Term Memory Network - LSTM) kullanılmıştır.

Önerilen modelin mimarisi Şekil 1’de verilmektedir. Önerilen model 3 katmandan oluşmaktadır: Girdi, LSTM katmanı ve çıktı katmanları. Girdi katmanı sözcük dizisini almakta, LSTM katmanı LSTM birimlerini kullanarak çıktıları hesaplamakta, çıktı katmanı ise olası sonraki sözcüğü tahmin etmektedir. Çıktı katmanında softmax aktivasyon fonksiyonu kullanılarak sözcük kümesinden en yüksek olasılığa sahip bir sonraki sözcük tahmin edilmeye çalışılmaktadır.

A. Veri Kümesi

Şarkı sözü üretmek için kullandığımız modeli eğitmek için Selda Bağcan’ın şarkıları kullanılmıştır. Hazır bir veri kümesi olmadığından şarkı sözleri İnternette manuel yöntemlerle elde edilmiştir (web crawling). Oluşturduğumuz veri kümesinde toplam 222 şarkı ve bu şarkılara ait 16273 sözcük bulunmaktadır.

B. Önışleme

Veri kümesi üzerinde önışlem olarak, bütün şiirlerin başına şiir başı (<S>) ve sonuna şiir sonu (</S>) terimleri eklenmiş ve noktalama işaretleri şiirden çıkarılmıştır. Şiirler terimlerine ayrılmıştır (tokenization).

C. HiperParametreler

Modelin girdi katmanında sözcük vektörleri kullanılmaktadır. Sözcük vektörleri olarak, word2vec [19] ile Boun veri kümesi [20] üzerinde önceden eğitilmiş 200 boyutlu vektörler kullanılmıştır. Kullanılan hiperparametreler Tablo I’de verilmiştir.

TABLO I: Modelin eğitilmesinde kullanılan parametreler

Sözcük vektörleri	word2vec [19] 200d
LSTM katman sayısı	1
İyileştirici (Optimizer)	Adam [21]
Seyreltme parametresi (Dropout)	0.3
Saklı katman boyutu (Hidden size)	32
İterasyon (Epoch)	150

D. Sonuçlar

Dil modelleri için standart değerlendirme metriği olarak çapraşıklık (perplexity) kullanılmaktadır. Çapraşıklık matematiksel olarak aşağıdaki gibi ifade edilmektedir:

$$PP = \prod_{t=1}^T \left(\frac{1}{P_{LM}(w_n | w_1, w_2, \dots, w_{n-1})} \right)^{1/T} \quad (1)$$

w_1, \dots, w_n cümledeki sözcüklere, T cümledeki sözcük sayısına ve P_{LM} cümlelerin dil modeli sonucu elde edilen olasılığına karşılık gelmektedir.

Eğittiğimiz model ile ürettiğimiz 30 şarkı için ortalama perplexity değeri 35.3 olarak bulunmuştur. Ürettiğimiz şarkılara bazı örnekler aşağıda verilmiştir:

gayrı dayanmam ben bu hasrete
bol gibi gözlerinden akan yaş niye
bu suskunluk bu durgunluk sıkıntın niye
çok uzakta öyle bir yer var
ne yar verdin bir gün mü yolundan

dağlar derviş
rabbini görmüş
hey oğlum memiş
sabır et
sabır et sabır et

IV. NÖRAL DİL MODELİ İLE MELODİ ÜRETME

Melodi üretmek için benzer şekilde LSTM tabanlı bir nöral dil modeli oluşturulmuştur. Bu çalışmada kullanılan veri kümesi MIDI (Musical Instrument Digital Interface – Müzik Enstrümanları Dijital Arabirimi) formatında ve enstrüman olarak piyano ile oluşturulan müzikleri içermektedir. Midi dosyalarının içeriğinde müziğe ait notalar ve bu notaların bazı özellikleri yer almaktadır.

A. Veri Kümesi

Kullanılan veri kümesi, 20 adet pdf (Portable Document Format - Taşınabilir Belge Biçimi) formatına gömülü notalardan oluşan Türk Halk Müziği parçaları içermektedir. Bu parçalar IMSLP müzik kütüphanesinden elde edilmiştir³. Projede midi dosyası kullanıldığından elde edilen parçalar, öncelikle midi formatına dönüştürülmüştür. Bu dönüştürme sonucunda 105 adet midi formatında nota kümesi elde edilmiştir.

B. Önışleme

Çalışmada, midi formatındaki dosyaları okumak, notaları, akorları ve müziğe ait bazı özellikleri elde etmek için,

³https://imslp.org/wiki/Main_Page

"pretty_midi" kütüphanesi ⁴ kullanılmıştır. Bu kütüphane kullanılarak, müzik parçalarına ait notalar, her iki nota arasındaki süre, piyanoda notaya basılma şiddeti değerleri (velocity) ve her notanın uzunluğu elde edilmiştir.

Midi dosyaları okunduktan sonra nota değerlerinin işlenebilmesi için öncelikle notalar stringe çevrilmiştir. Stringe çevrilen nota değerlerinin tekrarları silinmiştir. Notaların sinir ağına girdi olarak uygun olması için stringler sayısal değerlere çevrilmiştir. Her bir nota değeri tek sıcak vektörüne (one hot encoding) dönüştürülmüş ve LSTM için giriş sekansları oluşturulmuştur. Bu şekilde her bir nota veya akor girdisi için bir sonraki nota veya akor tahmini LSTM tarafından gerçekleştirilmiştir.

Her sekansın uzunluğu 100 nota/akor olarak belirlenmiştir. Böylece, dizideki bir sonraki notanın tahmini için LSTM'in önceki 100 nota bilgisine de sahip olması gerekmektedir. Bu uzunluk, farklı sekans uzunlukları ile model test edildikten sonra belirlenmiştir.

Örnek Nota ve akorlar aşağıda verilmiştir:

```
A#5-D6
A#5-F#5
A2
A2-A1
A2-A3
B4-D#5-F4-G#4
B4-D5
B4-D5-E5-G5-B5
B4-E5-G#5-B5
B4-E5-G5
C#3-E3-F#3-A#3
```

Burada, iki nota arasındaki "-" işareti basılan notaların akor olduğunu, "#" sembolü basılan notanın minör veya major nota olduğunu belirtmektedir. Notaların yanındaki rakamlar ise notanın hangi oktavda basıldığını belirtir.

C. Model

Modelde kullanılan özellikler notalara ait frekans (pitch) aralıklarıdır. Bunun haricinde nota uzunluğu gibi özelliklere (feature) sabit değerler atanmış, böylece özellik kümesi mümkün olduğunca küçük tutulmaya çalışılmıştır. Notaya ait süre gibi diğer özellikler eklendiğinde, veri kümesinin kısıtlı olmasından ötürü öğrenmeyi negatif olarak etkilediği gözlenmiştir. Sonuçta 258 tane özellik elde edilmiştir. Her bir özelliğe ait tek sıcak vektörü 3 katmanlı LSTM ağına verilmiştir. LSTM ağı 128 yığın (batch) boyutu ile 128 iterasyon (epoch) boyunca eğitilmiştir. Modele ait kodlar herkese açık olarak paylaşılmıştır⁵.

D. Nöral Dil Modeliyle Nota Üretim Aşaması

Üretilecek müziğin notalarını oluştururken, her seferinde farklı başlangıca sahip olan müzik üretmek için, üretilen ilk nota rastgele belirlenmiştir. Başlangıç notasının farklı olması üretilen müzikleri de birbirlerinden farklı kılacaktır. Üretilen

notaları birleştirmek için "pretty_midi" kütüphanesi kullanılmıştır. Notalara, bekleme süreleri, piyanoda klavyeye basma şiddeti (velocity) ve notaların uzunluk değerleri eklenmiştir.

Üretilen nota dizilerine örnekler:

```
Note(start=0.0000, end=0.5000, pitch=51, velocity=110)
Note(start=0.5000, end=1.0000, pitch=59, velocity=110)
Note(start=1.0000, end=1.5000, pitch=41, velocity=110)
Note(start=1.5000, end=2.0000, pitch=38, velocity=110)
Note(start=1.5000, end=2.0000, pitch=50, velocity=110)
Note(start=2.0000, end=2.5000, pitch=38, velocity=110)
Note(start=2.0000, end=2.5000, pitch=50, velocity=110)
Note(start=2.5000, end=3.0000, pitch=50, velocity=110)
```

Üretilen notalar, yan yana eklenerek txt uzantılı metin dosyası formatında düzenlenmiştir. Örnek bir metin dosyası aşağıda verildiği gibidir:

```
C4 G4 A#3 D4 G4 C4 D#4 G4 A#3 D4 G4 G3 C4 G4 A#3 D4
G4 C4 D#4 G4 A#3 D4 G4 G#3 C4 G4 A#3 D4 G4 C4 D#4 G4
A#3 D4 G4 G#3 C4 G4 G3 D4 G4 C4 D#4 G4 A#3 D4 G4 G#3 C4
G4 A3 C4 D#4 D#3 A#3 D#4 F3 C4 F4 G3 D4 G4 G4 C4 D#4 G4
A#3 D4 G4 G#3 C4 G4 A#3 D4 G4 C4 D#4 G4 A#3 D4 G4 G#3
C4 G4 G3 D4 G4 C4 D#4 G4 A#3
```

V. TÜRKÇE ŞARKI SÖZÜ SENTEZLEME

Çalışmanın son aşamasında, üretilen şarkı sözleri ile üretilen notalar sentezlenerek şarkıların verilen notalarla seslendirilmesi işlemi gerçekleştirilmiştir. Şarkı sentezleme için açık kaynak kodlu eCantorix [22] kütüphanesi ve konuşma sentezleme için ise yine açık kaynak kodlu olan ve Türkçe desteği de bulunan eSpeak [23] kütüphanesi kullanılmıştır. eSpeak "Formant sentezi" metodunu kullanmaktadır. Bu, birçok dilin küçük boyutlarda oluşturulmasını sağlamaktadır. eSpeak tamamen bilgisayar tarafından üretilen sesi kullandığı için, insan seslerinden oluşturulan daha büyük sentezleyiciler kadar doğal ve pürüzsüz değildir.

Öncelikle şarkı sözleri TurkishNLP [24] kütüphanesi kullanılarak hecelerine ayrılmıştır. Böylece notalar ile sözler eşleştirildiğinde heceler birbiriyle uyumlu bir şekilde seslendirilebilmiştir.

Sözcükler hecelerine ayrıldıktan sonra, her bir hecenin bir nota ile eşleştirilmesi işlemi yapılmıştır. Bu kısımda ".abc" dosyası ve sesin kalınlık, incelik ve dilinin verildiği ".conf" dosyası oluşturulmaktadır. ".abc" dosyasının içeriğinde notalar ve o notalara karşılık gelen sözler yer almaktadır. eCantorix, ".abc" ve ".conf" dosyalarını birleştirerek seslendirilmiş olan ".wav" dosyalarını çıktı olarak vermektedir. Aşağıda örnek ".abc" ve ".conf" dosyaları verilmiştir.

Örnek ".conf" dosyası:

```
$ESPEAK_VOICE = "tr";
$ESPEAK_TRANSPOSE = -15;
do ecantorix/examples/extravoices/melt.inc
```

".conf" dosyasındaki "transpose" değeri sesin kalınlık ve inceliğini ayarlar. "Transpose" değeri sıfıra yaklaştıkça ses incelik, uzaklaştıkça ses kalınlığıdır.

⁴<https://github.com/craffel/pretty-midi>

⁵<https://github.com/haywiree/AI-Music-Composer>

Örnek ".abc" dosyası:

```
X:0
M:4/4
L:1/4
Q:120
K:C
V:1
B4 C#5 E5 A4 C5 B4 C#5 C#5 G#4 |
w:sev-gi-,a-cı-yı öğ-ren-mek-tir
B4 C5 D#5 C#5 B4 A4 C5 B4 D5 C5 |
w:tüm ben-cil-lik-ler-den iğ-ren-mek-tir
E5 D#5 B5 F#4 A#5 B3 A#4 F#4 G#4 G#4 D#4 |
w:bir öz-ge kur-ban-lı-ğa o-lup ta-lip
A4 G#4 D#4 A#4 B4 A#4 G#4 F#4 F4 D#4 |
w:her an,-her sa-ni-ye doğ-ran-mak-tır
```

".abc" dosyası, bir başlık (header) içerir. Burada verilen, X referans numarasıdır. İlk tonun referansı X: 1 olmalıken, ikinci tonun referans numarası X: 2 olmalıdır. M metredir. Sayaç hakkındaki bilgiler üç yoldan biriyle girilebilir. Birincisi standart - M: 6/8 veya M: 3/4. Ayrıca özel semboller M: C ve M: C | ortak zamanı ve kesme zamanını temsil eder. Son olarak, karmaşık metreler M: (2 + 3 + 2) / 8 - formatında belirtilebilir. L birim nota uzunluğudur, yani bir abc dosyasında tek bir harfin hangi notayı temsil ettiğini gösterir. Bu nota uzunluğu için standart değerleri kullanır - L: 1/4 çeyrek nota veya dörtlük, L: 1/8 sekizinci nota veya sekizlik, vb. Q, dakikadaki vuruş sayısı olarak tempodur. En basit biçimde, bu Q: 1/2 = 120 (dakikada 120 yarım nota vuruş) gibi bir değer olacaktır, ancak tanım dört adede kadar atım içerebilir; Q: 1/4 3/8 1/4 3/8 = 40. K anahtardır. Bu alanın ilk oluşumu her zaman header bölümünün sonunu gösterir. Anahtar alanın temel yapısı aşağıdaki gibidir: A ve G arasındaki büyük harf (tuş imzası), sırasıyla keskin veya düz (isteğe bağlı), modu belirtmek için veya b (hiçbiri belirtilmezse, majör olduğu varsayılır). V dizedir. İlk dize V: 1 ile belirtilirken, ikinci dize V: 2 ile belirtilir. Şarkı sentezleme, "kalın", "normal" ve "ince" olmak üzere üç farklı ses tonuyla yapılabilmektedir.

Örnekte de görüldüğü üzere, ".abc" dosyasında önce notalar, ardından "w" başlığı ile hecelenmiş ve notalarla hizalanmış şarkı sözü satırları yer almaktadır.

Sonuç olarak üretilen şarkılar değerlendirildiğinde, notaların çok uzun ve kalın olmaları durumunda sonucun anlaşılmasının biraz zor olduğu gözlemlenmiştir. Eğer notalar daha kısa ve inceye yakın olursa ve şarkı sözleri de söylemesi kolay ve anlaşılabilir basit kelimelerden oluşursa daha başarılı sonuçlar alınabilmektedir. Şarkı sentezlemeye ait kodlar herkese açık olarak paylaşılmıştır⁶.

VI. SONUÇ

Bu çalışmada, Türkçe şarkı sentezleme için derin öğrenme kullanan modeller önerilmiştir. Bu amaçla, hem şarkı sözlerinin, hem de melodinin nöral dil modelleriyle eğitilmesi, hem de sonrasında sözlerin üretilen melodiyle sentezlenerek seslendirilmesi gerçekleştirilmiştir. Bilindiği kadarıyla daha önce Türkçe için herhangi bir şarkı sentezleme çalışması yapılmamıştır. Bundan dolayı, yapılan çalışmanın Türkçe müzik üretme çalışmalarına da öncülük edeceği düşünülmektedir.

Sonraki çalışmalarımıza temel olacak bu çalışmanın devamında, melodilerde sadece nota ve akor özelliklerini değil, bunun yanında notaların süresi gibi farklı özellikleri de önerilen modellere dahil etmeyi planlıyoruz.

KAYNAKLAR

- [1] L. A. Hiller and L. M. Isaacson, *Experimental Music; Composition with an Electronic Computer*. Westport, CT, USA: Greenwood Publishing Group Inc., 1979.
- [2] M. Balaban, K. Ebcioğlu, and O. Laske, Eds., *Understanding Music with AI: Perspectives on Music Cognition*. Cambridge, MA, USA: MIT Press, 1992.
- [3] D. Conklin and I. H. Witten, "Multiple viewpoint systems for music prediction," *Journal of New Music Research*, vol. 24, no. 1, pp. 51–73, 1995.
- [4] M. Balaban, K. Ebcioğlu, and O. Laske, Eds., *Understanding Music with AI: Perspectives on Music Cognition*. Cambridge, MA, USA: MIT Press, 1992.
- [5] D. Cope, *Computers and Musical Style*. Madison, WI, USA: A-R Editions, Inc., 1991.
- [6] M. Steedman, "A generative grammar for Jazz chord sequences," *Music Perception: An Interdisciplinary Journal*, vol. 2, 10 1984.
- [7] —, "The blues and the abstract truth: Music and mental models," *Mental models in cognitive science*, pp. 305–318, 1996.
- [8] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, 1st ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1989.
- [9] G. Wiggins, G. Papadopoulos, S. Phon-Amnuaisuk, and A. Tuson, "Evolutionary methods for musical composition," *In International Journal of Computing Anticipatory Systems*, 06 2000.
- [10] I. Zelinka, V. Snasael, and A. Abraham, *Handbook of optimization: from classical to modern approach*. Springer Science & Business Media, 2012, vol. 38.
- [11] P. M. Todd, "A connectionist approach to algorithmic composition," *Computer Music Journal*, vol. 13, no. 4, pp. 27–43, 1989.
- [12] D. Eck and J. Schmidhuber, "A first look at music composition using LSTM recurrent neural networks," Tech. Rep., 2002.
- [13] J. Franklin, "Recurrent neural networks for music computation," *INFORMS Journal on Computing*, vol. 18, pp. 321–338, 08 2006.
- [14] T. Liu and B. Ramakrishnan, "Bach in 2014: Music composition with recurrent neural network," *CoRR*, vol. abs/1412.3191, 2014.
- [15] C.-C. Chen and R. Miikkulainen, "Creating melodies with evolving recurrent neural networks," in *IJCNN'01. International Joint Conference on Neural Networks. Proceedings (Cat. No. 01CH37222)*, vol. 3. IEEE, 2001, pp. 2241–2246.
- [16] P. Potash, A. Romanov, and A. Rumshisky, "Ghostwriter: Using an LSTM for automatic Rap lyric generation," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, pp. 1919–1924.
- [17] —, "Ghostwriter: Using an LSTM for automatic Rap lyric generation," 01 2015, pp. 1919–1924.
- [18] A. C. T. Fernandez, K. J. M. Tarnate, and M. Devaraj, "Deep rapping: Character level neural models for automated Rap lyrics composition."
- [19] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.
- [20] H. Sak, T. Güngör, and M. Saraçlar, "Turkish language resources: Morphological parser, morphological disambiguator and web corpus," in *International Conference on Natural Language Processing*. Springer, 2008, pp. 417–427.
- [21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
- [22] R. Polzer, "Singing speech synthesis using eSpeak," 2012.
- [23] J. Duddington, "Speak speech synthesizer," 1995.
- [24] M. Cetinkaya, "Singing speech synthesis using eSpeak," 2018.

⁶<https://github.com/abdullahkokbiyik/robot-singer>