CHAPTER VI

ADVERSARIAL LEARNING FOR PIANO TRANSCRIPTION

In the previous chapter, we have introduced a WaveNet-based music synthesis model that is conditioned on predicted Mel spectrograms. A Mel spectrogram is a representation of audio whose dimension is in the order of hundreds at each frame. Accurate prediction of such high-dimensional representations requires sophisticated modeling of their statistical properties as well as extensive computational power to realize it. Piano roll representations of music, which are formulated in Chapter I as the output representation of polyphonic music transcription throughout this thesis, are another example of high-dimensional representations of audio which contain hundreds of data points at each instant. Therefore, unlike the single-valued objective in monophonic pitch estimation tasks, prediction of piano roll representations is inherently a multi-label problem and thus warrants a mathematical model that is right for high-dimensional representation.

In this chapter, as hinted in the conclusions in the last chapter, we discuss a method to include adversarial loss to allow the model to predict the piano roll target more accurately. We first address a limitation in the conventional, element-wise definition of loss functions in which the inter-label probabilistic dependencies are not accurately modeled. Based on this observation, we show that appending an adversarial discriminator to a discriminative piano transcription model can help producing more confident predictions which in turn improve the transcription

accuracy. The content of this chapter is largely based on the work presented at ISMIR 2019 (J. W. Kim & Bello, 2019).

1 Introduction

Automatic music transcription (AMT) is a multifaceted problem and comprises a number of subtasks, including multi-pitch estimation (MPE), note tracking, instrument recognition, rhythm analysis, score typesetting, etc. MPE predicts a set of concurrent pitches that are present at each instant, and it is closely related to the task of note tracking, which predicts the onset and offset timings of every note in audio. In this chapter, we address an issue in the recent approaches for MPE and note tracking, where the probabilistic dependencies between the labels are often overlooked.

A common approach for MPE and note tracking is through the prediction of a two-dimensional representation that is defined along the time and frequency axes and contains the pitch tracks of notes over time. Piano rolls are the most common example of such representations, and deep salience (Bittner et al., 2017) is another example that can contain more granular information on pitch contours. Once such representation is obtained, pitches and notes can be decoded by thresholding (Kelz et al., 2016) or other heuristic methods (J. W. Kim et al., 2018; Hawthorne et al., 2018).

To train a model that predicts a two-dimensional target representation $\hat{\mathbf{Y}} \in \mathbb{R}^{P \times T}$ from an input audio representation $\mathbf{X}$, where P is the number of pitch labels and T is the number of time frames, a common approach is to minimize the

element-wise sum of a loss function $\mathcal{L}$:

$$\text{minimize } \mathcal{L}(\hat{\mathbf{Y}}, \mathbf{Y}) = \sum_{p=1}^{P} \sum_{t=1}^{T} \mathcal{L}(\hat{\mathbf{Y}}_{pt}, \mathbf{Y}_{pt}), \tag{23}$$

where $\mathbf{Y} \in \mathbb{R}^{P \times T}$ is the ground truth. In a probabilistic perspective, we can interpret $\mathcal{L}$ as the negative log-likelihood of the model parameters ϑ of a discriminative model $p_\vartheta(\mathbf{Y}|\mathbf{X})$:

$$p_\vartheta(\mathbf{Y}|\mathbf{X}) = e^{-\mathcal{L}(\hat{\mathbf{Y}}, \mathbf{Y})} = \prod_{p=1}^{P} \prod_{t=1}^{T} e^{-\mathcal{L}(\hat{\mathbf{Y}}_{pt}, \mathbf{Y}_{pt})} = \prod_{p=1}^{P} \prod_{t=1}^{T} p_\vartheta(\mathbf{Y}_{pt}|\mathbf{X}) \tag{24}$$

which indicates that each element of the label $\mathbf{Y}$ is conditionally independent with each other given the input $\mathbf{X}$. This encourages the model to predict the average of the ground-truth distribution, making blurry predictions when the target is multimodal, e.g. natural images (Dosovitskiy & Brox, 2016).

Music data is highly contextual and multimodal, and the conditional independence assumption does not hold in general. This is why many computational music analysis models employ a separate post-processing stage after sequence prediction. One approach is to factorize the joint probability using the chain rule and assume the Markov property:

$$p_\vartheta(\mathbf{Y}|\mathbf{X}) \approx \prod_{p=1}^{P} \prod_{t=1}^{T} p_\vartheta(\mathbf{Y}_{pt}|\mathbf{Y}_{\cdot(t-1)}, \mathbf{X}). \tag{25}$$

This corresponds to appending hidden Markov models (HMMs) (Poliner & Ellis, 2006) or recurrent neural networks (RNNs) (Sigtia et al., 2016; Hawthorne et al., 2018) to the transcription model. The Markov assumption is effective for one-dimensional sequence prediction tasks, such as chord estimation (Ni

et al., 2012) and monophonic pitch tracking (Mauch & Dixon, 2014), but when predicting a two-dimensional representation, it still does not address the inter-label dependencies along the frequency axis.

There exist a number of models in the computer vision literature that can express inter-label dependencies in two-dimensional predictions, such as the neural autoregressive distribution estimator (NADE) (Larochelle & Murray, 2011), PixelRNN (van den Oord, Kalchbrenner, & Kavukcuoglu, 2016), and PixelCNN (van den Oord, Kalchbrenner, Vinyals, et al., 2016). However, apart from a notable exception using a hybrid RNN-NADE approach (Boulanger-Lewandowski et al., 2012), the effect of learning the joint multi-label distribution for polyphonic music transcription has not been well studied.

To this end, we propose a new approach for effectively leveraging inter-label dependencies in polyphonic music transcription. We pose the problem as an image translation task and apply an adversarial loss incurred by a discriminator network attached to the baseline model. We show that our approach can consistently and significantly reduce the transcription errors in *Onsets and Frames* (Hawthorne et al., 2018), a state-of-the-art music transcription model.

2 Background

2.1 Automatic Transcription of Polyphonic Music

Automatic transcription models for polyphonic music can be classified into frame- or note-level approaches. Frame-level transcription is synonymous with multi-pitch estimation (MPE) and operates on tiny temporal slices of audio, or frames, to predict all pitch values present in each frame. Note-level transcription, or note tracking, operates at a higher level, predicting a sequence of note events

that contains the pitch, the onset time, and optionally the offset time of each note. Note tracking is typically implemented as a post-processing stage on the output of MPE (Benetos et al., 2019), by connecting and grouping the pitch estimates over time to produce discrete note events. In this sense, we can say that MPE is at the core of polyphonic music transcription.

Among the approaches for MPE reviewed in Section II.3, two categories have been most successful in recent years: matrix factorization (Lee & Seung, 2001) and deep learning (LeCun et al., 2015). Commonly in both of these approaches, an iterative gradient-descent algorithm is used to minimize an element-wise loss function that is defined on two-dimensional representation. NMF-based methods are designed to minimize a divergence function between the matrix factorization and the target matrix, and similarly in deep learning models, neural networks are optimized to predict a two-dimensional representation that makes an element-wise loss function as small as possible.

In this chapter, we use Onsets and Frames (Hawthorne et al., 2018), a state-of-the-art piano transcription model based on deep learning, as our baseline. It uses multiple columns of convolutional and recurrent neural network layers to predict onsets, offsets, velocities, and frame labels from the Mel spectrogram input, as shown in Figure 18. Predicted onset and frame activations are then used for decoding the note sequences, where a threshold value is used to create binary onset and frame activations, and frame activations without the corresponding onsets are disregarded. Onsets and Frames also uses an element-wise optimization objective which does not consider the inter-label dependencies. This motivates the adversarial training scheme that is outlined in the following subsection.

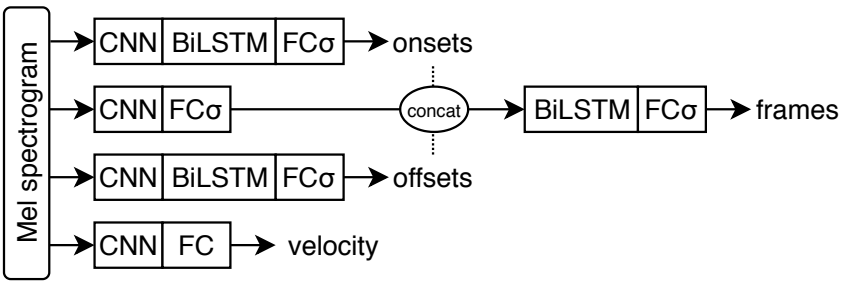

Figure 18: The Onset and Frames model. CNN denotes the convolutional acoustic model taken from (Kelz et al., 2016), FC denotes a fully connected layer, and σ denotes sigmoid activation. Dotted lines mean stop-gradient, i.e. no backpropagation.

2.2 Generative Adversarial Networks and `pix2pix`

As extensively reviewed in Section III.4, generative adversarial networks (GANs) (Goodfellow et al., 2014) refer to a family of deep generative models which consist of two components, namely the generator G and the discriminator D. Given a data distribution $\mathbf{x} \sim p(\mathbf{x})$ and latent codes $\mathbf{z} \sim p(\mathbf{z})$, GAN performs the following minimax game:

$$\min_G \max_D \underbrace{\left[\mathbb{E}_{\mathbf{x}} \log D(\mathbf{x}) + \mathbb{E}_{\mathbf{z}} \log(1 - D(G(\mathbf{z})))\right]}_{\mathcal{L}_{\text{GAN}}(G,D)}. \tag{26}$$

G and D are implemented as neural networks trained in an adversarial manner, where the discriminator learns to distinguish the generated samples from the real data, while the generator learns to produce realistic samples to fool the discriminator. GANs are most renowned for their ability to produce photorealistic images (Karras et al., 2019) and have shown promising results on music generation as well (Engel et al., 2019; Dong et al., 2017; Yang et al., 2017). We refer the readers to (Goodfellow, 2016; Creswell et al., 2017) for a comprehensive review of the techniques, variants, and applications of GANs.

The second term in Equation 26 has near-zero gradients when $D(G(\mathbf{z})) \approx 0$,

which is usually the case in early training. To avoid this, a *non-saturating* variant of GAN is suggested in (Goodfellow et al., 2014) where the generator is trained with the following optimization objective instead:

$$\max_G \ \mathbb{E}_{\mathbf{z}} \log D(G(\mathbf{z})). \tag{27}$$

The non-saturating GAN loss is used more often than the minimax loss in Equation 26 and is implemented by flipping the labels of fake data while using the same loss function. *Least-squares GAN* (Mao et al., 2017) is an alternative method to address the vanishing gradient problem, which replaces the cross entropy loss in Equations 26-27 with squared errors:

$$\begin{aligned} &\min_D \ \left[\mathbb{E}_{\mathbf{x}}(D(\mathbf{x})-1)^2 + \mathbb{E}_{\mathbf{z}} D(G(\mathbf{x}))^2\right], \\ &\min_G \ \mathbb{E}_{\mathbf{z}}(D(G(\mathbf{z}))-1)^2. \end{aligned} \tag{28}$$

The non-saturating GAN (NSGAN) and least-squares GAN (LSGAN) losses are examples of GAN losses that define the objective used during the minimax optimization, and we will use these two losses in conjunction with the conditional GAN setup described below.

While the default formulation of GAN concerns unconditional generation of samples from $p(\mathbf{x})$, conditional GANs (cGAN) (Mirza & Osindero, 2014) produce samples from a conditional distribution $p(\mathbf{y}|\mathbf{x})$. To do this, the generator and the discriminator are defined in terms of the condition variable $\mathbf{x}$ as well:

$$\min_G \max_D \underbrace{\left[\mathbb{E}_{\mathbf{x},\mathbf{y}} \log D(\mathbf{x},\ \mathbf{y}) + \mathbb{E}_{\mathbf{x},\mathbf{z}} \log(1 - D(\mathbf{x},\ G(\mathbf{x},\ \mathbf{z}))\right]}_{\mathcal{L}_{\text{cGAN}}(G,D)}. \tag{29}$$

`pix2pix` (Isola et al., 2017) is an image translation model that learns a mapping between two distinct domains of images, such as aerial photos and maps. A `pix2pix` model takes paired images $(\mathbf{x}, \mathbf{y})$ as its training data and minimizes the conditional GAN loss along with an additional L1 loss:

$$\mathbb{E}_{\mathbf{x},\mathbf{y},\mathbf{z}} \left\| \mathbf{y} - G(\mathbf{x}, \mathbf{z}) \right\|_1, \tag{30}$$

which encourages the conditional generator to learn the forward mapping from $\mathbf{x}$ to $\mathbf{y}$. It can be thought that the GAN loss in Equation 29 is fine-tuning the mapping learned by the L1 loss in Equation 30, resulting in a predictive mapping that better respects the probabilistic dependencies within the labels $\mathbf{y}$.

We adapt this approach to music transcription tasks and show that we can indeed improve the performance by introducing an adversarial loss to an existing music transcription model.

3 Method

We describe a general method for improving an NN-based transcription model G that performs prediction of a two-dimensional target $\mathbf{Y}$ from an input audio representation $\mathbf{X}$. Say the original model G is trained by minimizing the loss $\mathcal{L}_{\text{task}}(G(\mathbf{X}), \mathbf{Y})$ between the predicted target $\hat{\mathbf{Y}} = G(\mathbf{X})$ and the ground-truth $\mathbf{Y}$. The main idea of our method is to adapt `pix2pix` (Isola et al., 2017) to this setup, by introducing an adversarial discriminator D during the training process. The adversarial training objective includes the conditional GAN loss $\mathcal{L}_{\text{cGAN}}$ (Equation 29):

$$\min_G \max_D \mathbb{E}_{\mathbf{X},\mathbf{Y}} \Big[\nu \mathcal{L}_{\text{task}}(G(\mathbf{X}), \mathbf{Y}) + \mathcal{L}_{\text{cGAN}}(G, D) \Big], \tag{31}$$

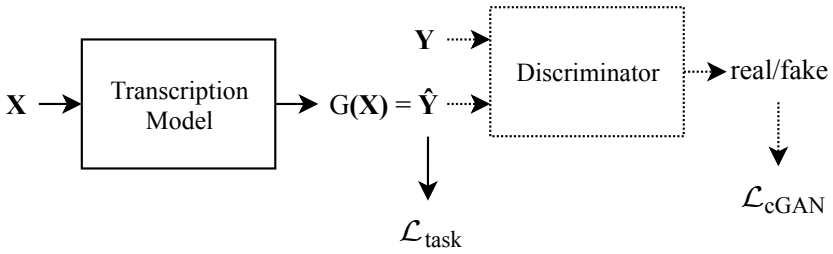

Figure 19: A computation graph showing how a discriminator is appended to the original model. The appended parts are shown as dotted components.

where ν is a hyperparameter that controls how much the conditional GAN loss contributes to the gradient steps relative to the discriminative loss $\mathcal{L}_{\text{task}}$. Figure 19 illustrates how the two components are connected in the computation graph and how the loss terms are calculated.

Adversarial training with $\mathcal{L}_{\text{cGAN}}$ allows the model to learn the inter-label dependencies as desired, even when $\mathcal{L}_{\text{task}}$ is defined only in terms of element-wise operations between $\hat{\mathbf{Y}}$ and $\mathbf{Y}$, as in Equation 23. In the next subsection, we describe a neural network architecture for the cGAN discriminator that leverages prior knowledge on music.

3.1 Musically Inspired Adversarial Discriminator

Following `pix2pix`, we use a fully convolutional architecture (Long et al., 2018) for the discriminator. By being fully convolutional, the discriminator has translation invariance not only along the time axis (as in HMMs and RNNs) but also along the frequency axis. Since the discriminator determines how realistic a polyphonic note sequence is, the translation invariance enforces that the decision does not depend on the musical key, but only on the relative pitch and time intervals between the notes. This effectively implements a music language model

(MLM) (Boulanger-Lewandowski et al., 2012; Sigtia et al., 2016) and biases the transcription toward more realistic note sequences.

Unlike the image-to-image translation problem, the input representations (e.g. Mel spectrograms) and the output representations (e.g. piano rolls) of a music transcription model can have different dimensions. This makes combining $\mathbf{X}$ and $\mathbf{Y}$ in a fully convolutional manner difficult. For this reason, we make the discriminator a function of $\mathbf{Y}$ only, simplifying the objective in Equation 29 to:

$$\mathcal{L}_{\text{cGAN}}(G, D) = \mathbb{E}_{\mathbf{Y}} \log D(\mathbf{Y}) + \mathbb{E}_{\mathbf{X}} \log(1 - D(G(\mathbf{X}))). \tag{32}$$

Note that $\mathbf{z}$ is also omitted in Equation 32, as we follow Isola et al. (2017) and implement the stochasticity of $\mathbf{z}$ only in terms of dropout layers (Srivastava et al., 2014), without explicitly feeding random noises into the generator. This causes a mode collapse problem where the learned $p(\mathbf{Y}|\mathbf{X})$ is not diverse enough, but it does not harm our purpose of producing more realistic target representations.

3.2 TTUR and *mixup* to Stabilize GAN Training

Although an ideal GAN generator can fully reconstruct the data distribution at the global optimum (Goodfellow et al., 2014), training of GANs in practice is notoriously difficult, especially for high-dimensional data (Goodfellow, 2016). This led to the inventions of a plethora of techniques for stabilizing GAN training, among which we employ the two-timescale update rule (TTUR) (Heusel et al., 2017) and *mixup* (H. Zhang et al., 2018). TTUR means simply setting the generator's learning rate a few times larger than that of the discriminator, which has been empirically shown to stabilize GAN training significantly.

The other technique, *mixup*, is an extension to empirical risk minimization where training data samples are drawn from convex interpolations between pairs of empirical data samples. For a pair of feature-target tuples $(\mathbf{X}_i, \mathbf{Y}_i)$ and $(\mathbf{X}_j, \mathbf{Y}_j)$ sampled randomly from the empirical distribution, their convex interpolation is given by:

$$\begin{aligned} \tilde{\mathbf{X}} &= \lambda \mathbf{X}_i + (1 - \lambda)\mathbf{X}_j \\ \tilde{\mathbf{Y}} &= \lambda \mathbf{Y}_i + (1 - \lambda)\mathbf{Y}_j \end{aligned} \tag{33}$$

where $\lambda \sim \text{Beta}(\alpha, \alpha)$, and α is the *mixup* hyperparameter which controls the strength of interpolation. When $\alpha = 0$, the Beta distribution becomes Bernoulli(0.5) which recovers the usual GAN training without *mixup*.

Input: Generator $G_\vartheta(\mathbf{X})$ with initial parameters ϑ, learning rate η, and loss function $\mathcal{L}_{\text{task}}(\hat{\mathbf{Y}}, \mathbf{Y})$, discriminator $D_\varphi(\mathbf{Y})$ with initial parameters φ, learning rate β, and loss function $\ell \in \{\text{BCE}, \text{MSE}\}$, batch size m, training data distribution $p(\mathbf{X}, \mathbf{Y})$, `pix2pix` weight ν, *mixup* strength α.
Output: Trained conditional generator $G_\vartheta(\mathbf{X})$.

---

while φ *and* ϑ *have not converged* **do**
  $\{(\mathbf{X}_i, \mathbf{Y}_i)\}_{i=1,\cdots,m} \leftarrow m$ samples from $p(\mathbf{X}, \mathbf{Y})$
  for $i = 1, \cdots, m$ **do**
    $\hat{\mathbf{Y}}_i \leftarrow G_\vartheta(\mathbf{X}_i)$
    $\lambda_i \leftarrow$ sample from Beta(α, α)
    $\tilde{\mathbf{Y}}_i \leftarrow \lambda_i \mathbf{Y}_i + (1 - \lambda_i)\hat{\mathbf{Y}}_i$
  end
  $\mathcal{L}^D_{\text{cGAN}} \leftarrow \sum_{i=1}^M \ell(D_\varphi(\tilde{\mathbf{Y}}_i), \lambda_i)$
  $\varphi \leftarrow \varphi - \beta \cdot \nabla_\varphi \mathcal{L}^D_{\text{cGAN}}$
  $\mathcal{L}^G_{\text{cGAN}} \leftarrow \sum_{i=1}^M \ell(D_\varphi(\tilde{\mathbf{Y}}_i), 1 - \lambda_i)$
  $\vartheta \leftarrow \vartheta - \eta \cdot \nabla_\vartheta \Big[\sum_{i=1}^m \nu \mathcal{L}_{\text{task}}(\hat{\mathbf{Y}}_i, Y_i) - \mathcal{L}^G_{\text{cGAN}} \Big]$
end

Algorithm 1: Training of a *mixup* Conditional GAN.

mixup is readily applicable to the binary classification task of GAN discriminators. In our conditional GAN setup, we have an additional advantage of having paired samples of a real label $\mathbf{Y}$ and a fake label $\hat{\mathbf{Y}} = G(\mathbf{X})$, which allow us to replace Equation 32 with:

$$\min_G \max_D \mathbb{E}_{\mathbf{X},\mathbf{Y},\lambda}\Big[- \ell(D(\lambda\mathbf{Y} + (1-\lambda)G(\mathbf{X})),\ \lambda)\Big]. \tag{34}$$

where $\ell(p, y) = -y \log p - (1-y)\log(1-p)$ is the binary cross entropy (BCE) function. With this *mixup* setup, the discriminator now has to operate on the convex interpolation between the predicted representation and the corresponding ground truth. This makes the discriminator's task even more difficult when the prediction gets close to the ground truth, which is desirable because the discriminator should be inconclusive (i.e. $D = \frac{1}{2}$ everywhere) at the global optimum (Goodfellow et al., 2014).

Algorithm 1 details the procedure of training the conditional GAN using *mixup*, based on Equations 32 and 34. Note that for training the generator network, we perform label flipping in $\mathcal{L}^G_{\text{cGAN}}$ similarly as in Equation 27. Also, to train a least-squares GAN (Equation 28) instead, we can simply replace ℓ with a mean squared error (MSE) loss.

4 Experimental Setup

To verify the effectiveness of our approach, we compare Onsets and Frames (Hawthorne et al., 2018), a state-of-the-art piano transcription model, with variants of the same model that are trained with the adversarial loss. We also aim to evaluate the choices of the GAN loss and the *mixup* strength α.

4.1 Model Architecture

We use the extended Onsets and Frames model (Hawthorne et al., 2019) which increased the CNN channels to 48/48/96, the LSTM units to 256, and the FC units to 768. The extended model has a total of 26.5 million parameters. We do not use the frame loss weights described in (Hawthorne et al., 2018) in favor of the offset stack introduced in the extended version (see Figure 18). During inference, we first calculate the activations corresponding to overlapping chunks of audio, with the same length as the training sequences, and perform overlap-add (OLA) using Hamming windows to obtain the full-length activation matrix. We perform OLA instead of applying the recurrent calculations for the full length, because the effects of adversarial learning are best achieved within the training sequence length.

The input to the discriminator has two channels for the onset and frame predictions. The discriminator has 5 convolutional layers: `c32k3s2p1`, `c64k3s2p1`, `c128k3s2p1`, `c256k3s2p1`, `c1k5s1p2`, where the numbers indicate the number of output channels, the kernel size, the stride amount, and the padding size. At each non-final layer, dropout of probability 0.5 and leaky ReLU activation with negative slope 0.2 are used. The mean of the final layer output along the time and frequency axes is taken as the discriminator output.

4.2 Hyperparameters

Table 4 summarizes the hyperparameters used during the experiments, which are mostly taken directly from Hawthorne et al. (2018) and Isola et al. (2017). Also following Hawthorne et al. (2018), we use Adam (Kingma & Ba, 2015) and apply learning rate decay of factor 0.98 in every 10,000 iterations, for both the generator

Hyperparameter	Values
Generator learning rate η	0.0006
Discriminator learning rate β	0.0001
Discriminator loss function ℓ	{BCE, MSE}
Batch size m	8
`pix2pix` weight ν	100
mixup strength α	{0, 0.2, 0.3, 0.4}
Activation threshold τ	0.5
Training sequence length	327,680

Table 4: Hyperparameters used during the experiments.

and the discriminator. We examine two types of GAN losses, the non-saturating GAN ($\ell = \text{BCE}$) and the least-squares GAN ($\ell = \text{MSE}$). For each GAN loss, multiple values of *mixup* strengths are compared with $\alpha = 0$, i.e. no *mixup*. Training runs for one million iterations, and the iteration that best performs on the validation set are used for evaluation on the test set.

4.3 Dataset

We use the MAESTRO dataset (Hawthorne et al., 2019), which contains Disklavier recordings of 1,184 classical piano performances. The dataset consists of 172.3 hours of audio, which are provided with 140.1, 15.3, and 16.9 hours of train/validation/test splits such that recordings of one composition only appear in the same split. We resample the audio to 16 kHz and down-mix into a single channel. Following Hawthorne et al. (2018), an STFT window of 2,048 samples is used for producing 229-bin Mel spectrograms, and a hop length of 32 ms is used. Training sequences sliced at random positions are used, unlike the official implementation which slices training sequences at silence or zero crossings.

4.4 Evaluation Metrics

The Onsets and Frames model perform both frame-level and note-level predictions, and their performance can be evaluated with the standard precision, recall, and F1 metrics. For multi-pitch estimation, we also report the error rate metrics defined by Poliner and Ellis (2006), which include total error, substitution error, miss error, and false alarm error. We use the `mir_eval` (Raffel et al., 2014) library for all metric calculations. For the note-level metrics, we use the default settings of the library, which use 50 ms for the onset tolerance, 50 ms or 20% of the note length (whichever is longer) for the offset tolerance, and 0.1 for the velocity tolerance.

5 Results

5.1 Comparison with the Baseline Metrics

Table 5 and 6 summarize the transcription performance, clearly showing a consistent improvement in the conditional GAN models over the Onsets and Frames baseline. Table 5 shows that both non-saturating GAN and least-squares GAN achieve the highest frame and note F1 scores when the *mixup* strength

			mixup strength α			
	Baseline	GAN type	0	0.2	0.3	0.4
Frame F1	0.899	Non-Saturating	0.664	0.912	**0.914**	0.907
		Least-Squares	0.904	0.903	0.906	0.898
Note F1	0.942	Non-Saturating	0.717	0.953	**0.956**	0.951
		Least-Squares	0.944	0.947	0.950	0.943

Table 5: Frame and note F1 scores are the highest when the non-saturating GAN loss and $\alpha = 0.3$ are used.

			Non-Saturating GAN			Least-Squares GAN		
		Baseline	$\alpha = 0.2$	$\alpha = 0.3$	$\alpha = 0.4$	$\alpha = 0.2$	$\alpha = 0.3$	$\alpha = 0.4$
Frame Metrics	F1	0.899	0.912	**0.914**	0.907	0.901	0.906	0.898
	Precision	0.946	0.937	0.931	0.939	0.940	0.942	0.946
	Recall	0.857	0.889	0.898	0.879	0.865	0.875	0.855
	E_{total}	0.179	0.157	0.156	0.166	0.176	0.167	0.181
	E_{subs}	0.013	0.013	0.012	0.013	0.014	0.013	0.013
	E_{miss}	0.130	0.097	0.089	0.108	0.121	0.113	0.132
	E_{fa}	0.036	0.047	0.054	0.045	0.042	0.042	0.036
Note	F1	0.942	0.953	**0.956**	0.951	0.944	0.950	0.941
	Precision	0.990	0.974	0.981	0.973	0.986	0.988	0.989
	Recall	0.899	0.933	0.932	0.930	0.905	0.916	0.898
Note with Offsets	F1	0.802	0.811	**0.813**	0.799	0.799	0.810	0.798
	Precision	0.842	0.828	0.835	0.817	0.835	0.841	0.838
	Recall	0.765	0.794	0.793	0.782	0.767	0.781	0.762
Note with Offsets and Velocity	F1	0.790	0.799	**0.802**	0.787	0.788	0.799	0.787
	Precision	0.830	0.816	0.823	0.805	0.823	0.830	0.827
	Recall	0.755	0.783	0.782	0.770	0.757	0.771	0.752

Table 6: Summary of transcription performance. The non-saturating GAN loss achieves the best performance across all F1 metrics. The average metrics across the tracks in the MAESTRO test dataset are reported, and the model checkpoint where the average of frame F1 and note F1 is the highest on the validation dataset is used.

$\alpha = 0.3$ is used, and they both outperform the baseline. The binary piano rolls are easy to distinguish from the non-binary predictions, which may cause imbalanced adversarial training. *mixup* allows non-binary piano rolls to be fed to the discriminator, making its task more challenging and leading to higher performance.

Table 6 shows an important trend of the cGAN results compared to the baseline that cGAN trades off a bit of precision for a significant improvement in recall; this is a side effect of the cGAN producing more confident predictions, as will be discussed in the following subsections.

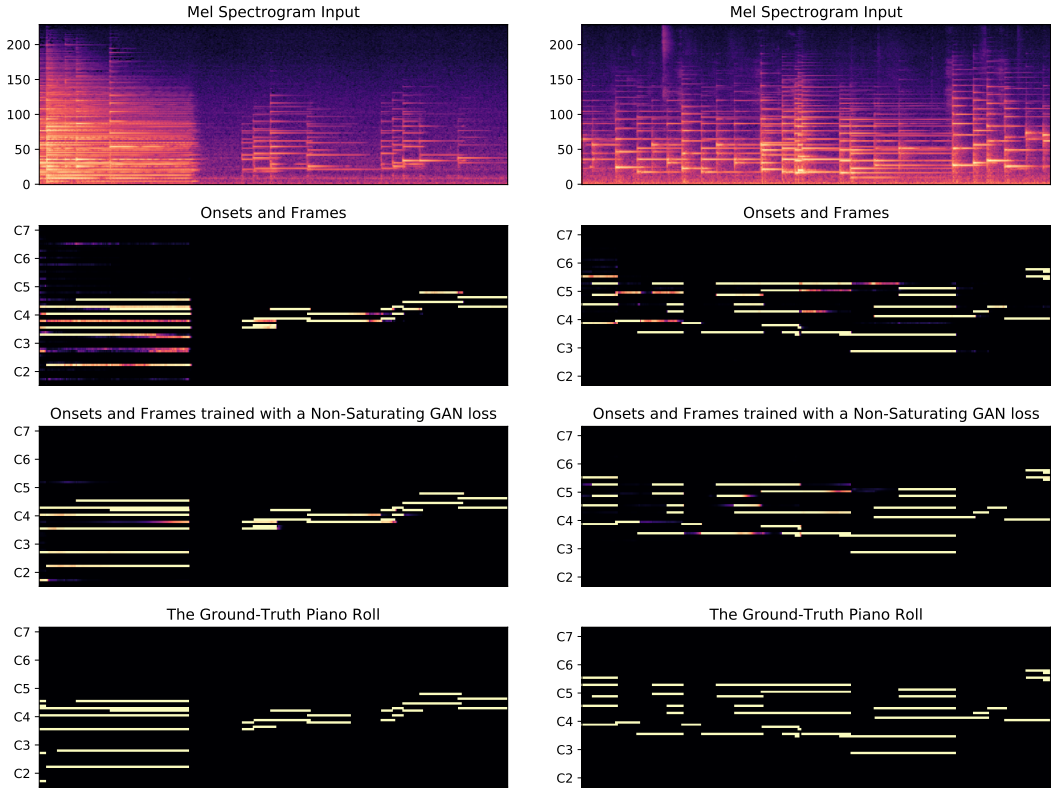

Figure 20: Comparisons of the frame activations predicted by the baseline and our model ($\ell = \text{BCE}$, $\alpha = 0.3$), on three example segments. The input Mel spectrograms and the target piano rolls are shown together. The GAN version produces more confident predictions compared to the noisy baselines, leading to more accurate predictions.

While the percentage differences are moderate, our method achieves statistically significant improvements in F1 metrics on the MAESTRO test dataset ($p < 10^{-14}$ for all 4 metrics, two-tailed paired t-test). The distribution of per-track improvement in each F1 metric is shown in Figure 21, which indicates that the improvements are evenly distributed across the majority of the tracks. These improvements are especially promising, considering that Onsets and Frames is already a very strong baseline.

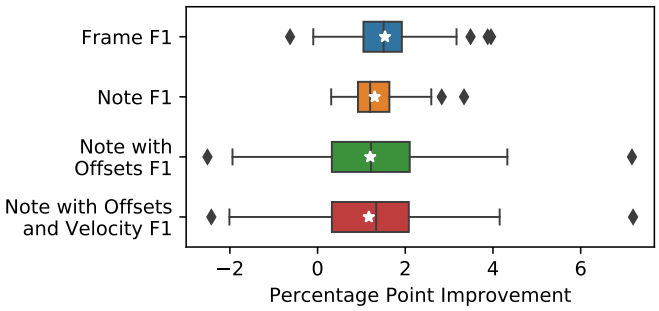

Figure 21: Distribution of the F1 score improvements over the baseline, tested on the MAESTRO test tracks.

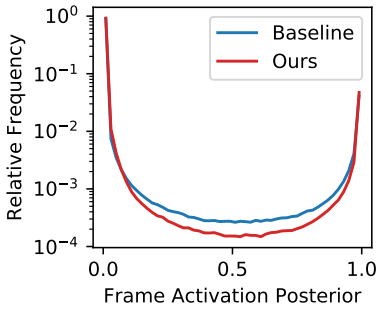

Figure 22: Distribution of frame activation values. Our model outputs more confident predictions, as indicated by the lower relative frequency in (0.1, 0.9).

5.2 Visualization of Frame Activations

To better understand the inner workings of the conditional GAN framework, we visualize the frame activations created by the baseline and the best performing conditional GAN model in Figure 20. In contrast to the baseline which have many blurry segments, the activations generated by our method mostly contain segments with solid colors, meaning that the model is more confident in its prediction. Figure 22 shows that the proportion of frame activation values in $(0.1, 0.9)$ is noticeably higher in the baseline, thus making the output less sensitive to the threshold choice. This is because indecisive predictions are penalized by the discriminator, since they are easy to distinguish from the ground-truth which

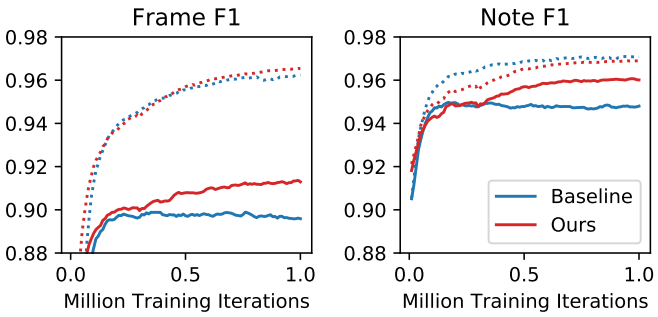

Figure 23: Learning curves showing the generalization gaps; training curves are drawn as dotted lines, and test curves are drawn as solid lines.

contains only binary labels. The generator is therefore encouraged to output the most probable note sequences even when it is unsure, rather than producing blurry activations that might hamper the decoding process. This allows for an interpretation in which the GAN loss provides a prior for valid onset and frame activations, and the model learns to perform MAP estimation based on this prior.

5.3 Training Dynamics and The Generalization Gap

Figure 23 shows the learning curves for the frame F1 and note F1 scores, where the scores on the training dataset are plotted in dotted lines. It is noticeable in the figure that the validation F1 scores for the baseline stagnate after 300k iterations, while the F1 scores of our model steadily grow until the end of 1 million iterations. Thanks to this, the generalization gap — the difference between the training and validation F1 scores — is significantly smaller for the conditional GAN model. This means that the GAN loss works as an effective regularizer that encourages the trained model to generalize better to unseen data, rather than memorizing the note sequences in the training dataset as LSTMs are known to be capable of (Zaremba et al., 2014).

6 Conclusions

We have presented an adversarial training method that can consistently outperform the baseline Onsets and Frames model, using the standard frame-level and note-level transcription metrics and visualizations that show how the improved model predicts more confident output. To achieve this, a discriminator network is trained competitively with the transcription model, i.e. a conditional generator, so that the discriminator serves as a learned regularizer that provides a prior for realistic note sequences. After training, the discriminator can be disregarded for inference, incurring no additional computational cost during transcription.

Our results show that modeling the inter-label dependencies in the target distribution is important and brings measurable performance improvements. Our method is generic, and any model that involves predicting two-dimensional representation should be able to benefit from including an adversarial loss. These approaches are common not only in transcription models but also in speech or music synthesis models that predict spectrograms as an intermediate representation (Shen et al., 2018; J. W. Kim et al., 2019). This implies that the findings in this chapter is applicable to a broader set of problems not only in multi-pitch estimation but also in the field of music information retrieval in general.

Our results do not include the effects of using data augmentation (Hawthorne et al., 2019), which is orthogonal to our approach and should bring additional performance improvements when applied. As discussed, the discriminator imposes the prior on the target domain whereas data augmentation enriches the input audio distribution. This implies that our method would be less effective

when the majority of errors are due to the discrepancy in the audio distribution between the training and test datasets. How to apply adversarial learning for better generalization on the input distribution is a potential future research direction.

We have argued that the adversarial discriminator serves as a regularizer providing the prior knowledge of how the note sequences in the dataset should look like, which complements the conditionally independent output of the Onsets and Frames model. The fully-connected output layer also does not take into account a very important characteristic of pitch, that each pitch corresponds to quasi-periodic signals in the input with a known frequency that are geometrically spaced — rather, the predictions are instead made as if each pitch is completely independent pieces of information, not utilizing the knowledge on pitch at all. Another important concept that is not considered in this chapter is the timbre. Although the MAESTRO dataset used in this chapter contains various recording conditions and therefore somewhat different timbres among the tracks, they are all recordings of piano performances which have limited timbral diversity. These aspects motivate building an improved model that can leverage the regularity of pitch as well as the knowledge of instrumental timbres. In the next chapter, we discuss how to extend our music transcription model to incorporate such aspects, specifically by using a synthesizer model similar to the one used in Chapter V.