2023 X:AI ADV Session
Toy Project 최종 발표

# Formula To Latex

**CV2 Team**
황건하 이승학 유광열 이서연

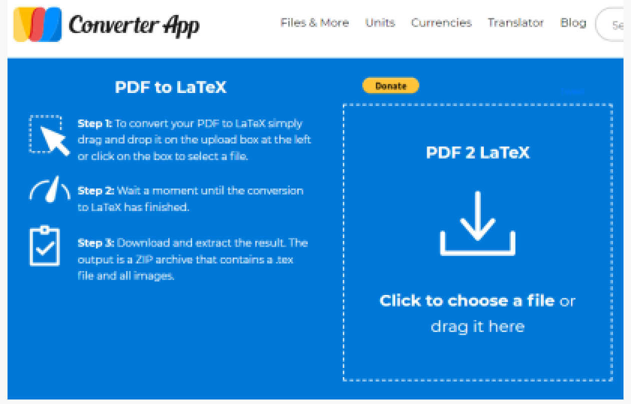
# Team Member

| 황건하 | 이승학 | 유광열 | 이서연 |
|---|---|---|---|
| 행정학전공 16 | AI빅데이터융합경영 18 | AI빅데이터융합경영 19 | AI빅데이터융합경영 21 |

# 연구 배경

1. 논문 번역 시 수식 변환의 어려움 존재

2. 기존에 존재하는 PDF to LaTex가 제대로 작동되지 않음



PDF to LaTeX

**Step 1:** To convert your PDF to LaTeX simply drag and drop it on the upload box at the left or click on the box to select a file.

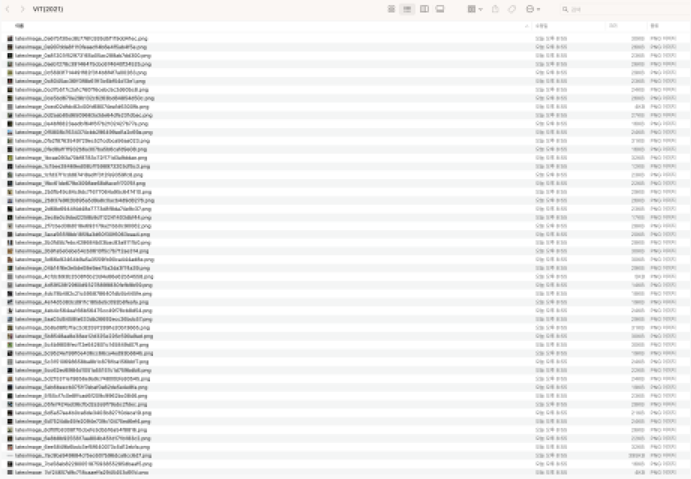
**Step 2:** Wait a moment until the conversion to LaTeX has finished.

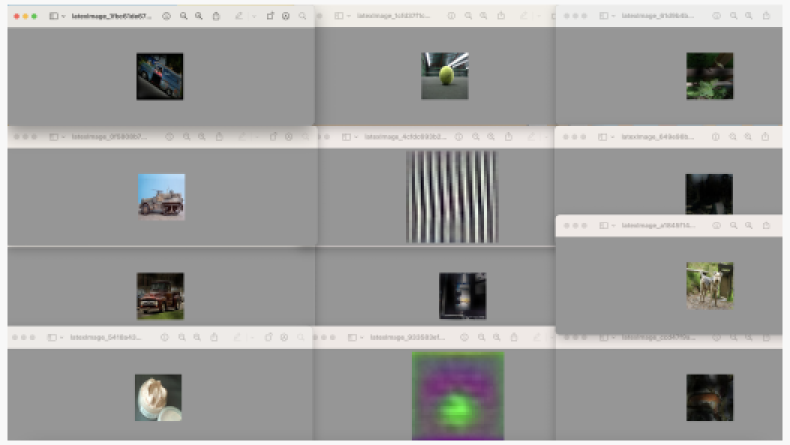
**Step 3:** Download and extract the result. The output is a ZIP archive that contains a .tex file and all images.

PDF 2 LaTeX

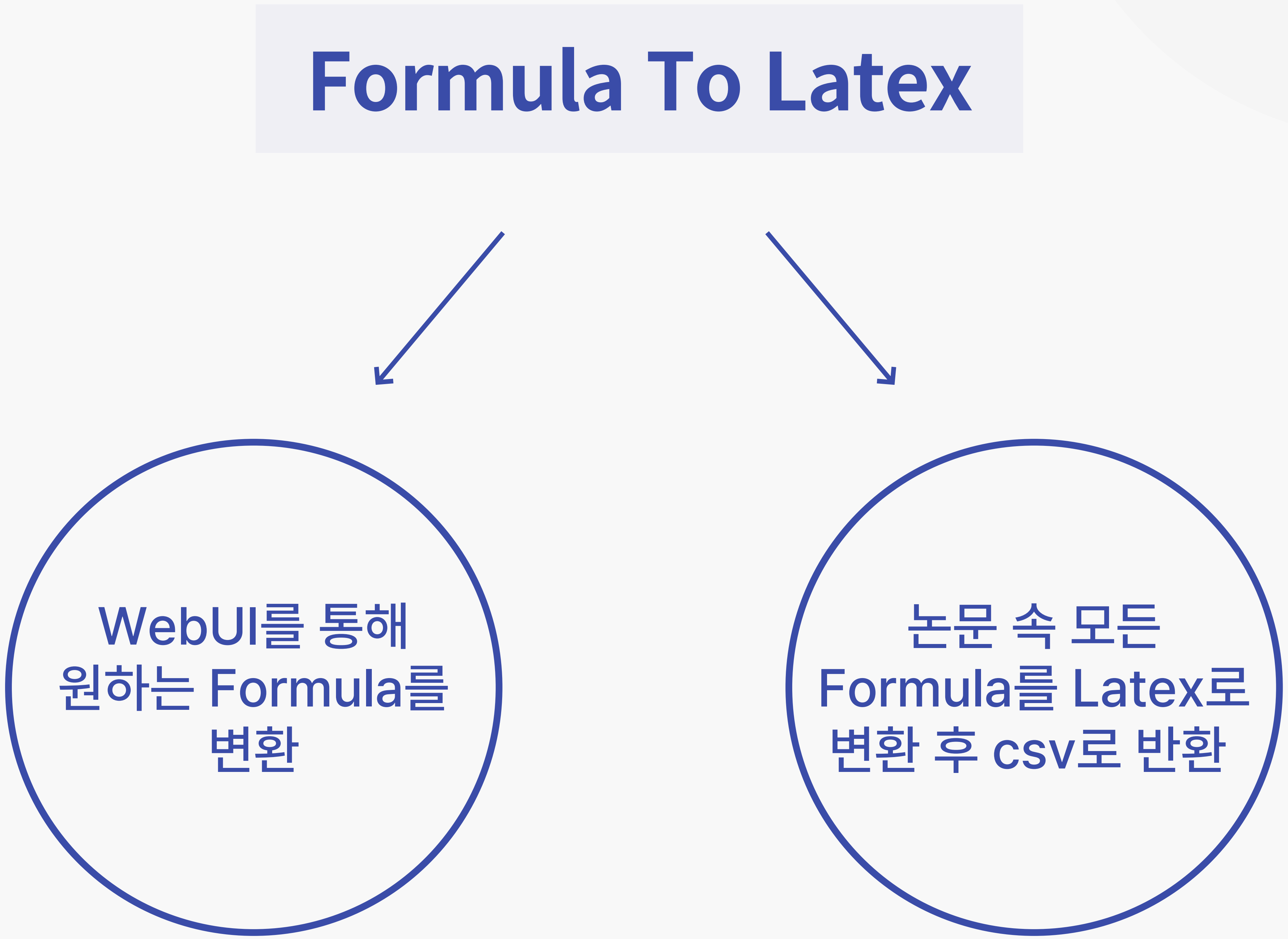
**Click to choose a file** or drag it here

▶▶





# 주제



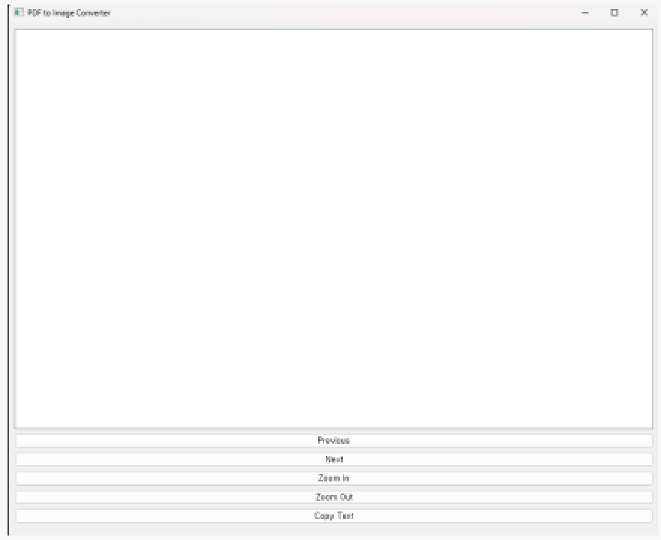
Formula To Latex

WebUI를 통해 원하는 Formula를 변환

논문 속 모든 Formula를 Latex로 변환 후 csv로 반환

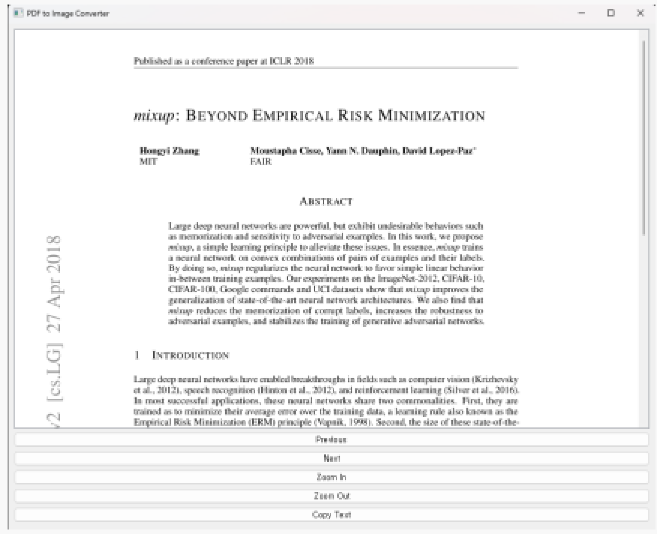
# Method1 - WebUI를 통해 원하는 Formula를 변환

## WebUI 실행



▶▶

## PDF 형식의 논문 드래그



Published as a conference paper at ICLR 2018

*mixup*: BEYOND EMPIRICAL RISK MINIMIZATION

Hongyi Zhang
MIT

Moustapha Cisse, Yann N. Dauphin, David Lopez-Paz*
FAIR

ABSTRACT

Large deep neural networks are powerful, but exhibit undesirable behaviors such as memorization and sensitivity to adversarial examples. In this work, we propose *mixup*, a simple learning principle to alleviate these issues. In essence, *mixup* trains a neural network on convex combinations of pairs of examples and their labels. By doing so, *mixup* regularizes the neural network to favor simple linear behavior in-between training examples. Our experiments on the ImageNet-2012, CIFAR-10, CIFAR-100, Google commands and UCI datasets show that *mixup* improves the generalization of state-of-the-art neural network architectures. We also find that *mixup* reduces the memorization of corrupt labels, increases the robustness to adversarial examples, and stabilizes the training of generative adversarial networks.

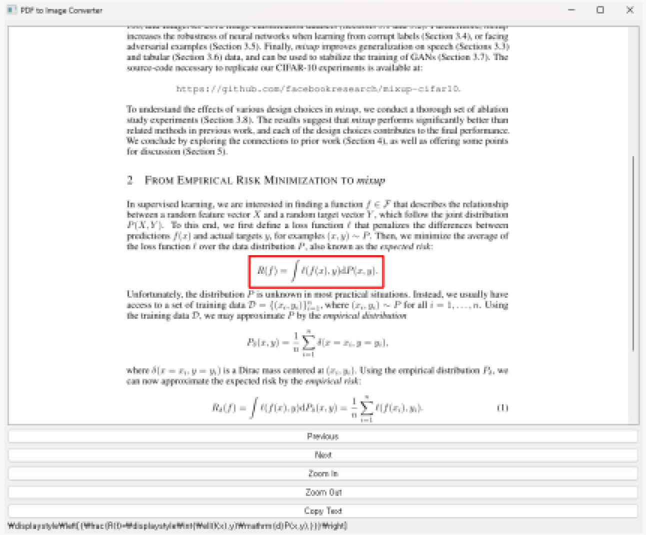
1 INTRODUCTION

Large deep neural networks have enabled breakthroughs in fields such as computer vision (Krizhevsky et al., 2012), speech recognition (Hinton et al., 2012), and reinforcement learning (Silver et al., 2016). In most successful applications, these neural networks share two commonalities. First, they are trained as to minimize their average error over the training data, a learning rule also known as the Empirical Risk Minimization (ERM) principle (Vapnik, 1998). Second, the size of these state-of-the-

v2 [cs.LG] 27 Apr 2018

▶▶

## 원하는 Formula 영역 지정



increases the robustness of neural networks when learning from corrupt labels (Section 3.4), or facing adversarial examples (Section 3.5). Finally, *mixup* improves generalization on speech (Sections 3.3) and tabular (Section 3.6) data, and can be used to stabilize the training of GANs (Section 3.7). The source-code necessary to replicate our CIFAR-10 experiments is available at:

https://github.com/facebookresearch/mixup-cifar10.

To understand the effects of various design choices in *mixup*, we conduct a thorough set of ablation study experiments (Section 3.8). The results suggest that *mixup* performs significantly better than related methods in previous work, and each of the design choices contributes to the final performance. We conclude by exploring the connections to prior work (Section 4), as well as offering some points for discussion (Section 5).

2 FROM EMPIRICAL RISK MINIMIZATION TO *mixup*

In supervised learning, we are interested in finding a function $f \in \mathcal{F}$ that describes the relationship between a random feature vector $X$ and a random target vector $Y$, which follow the joint distribution $P(X, Y)$. To this end, we first define a loss function $\ell$ that penalizes the differences between predictions $f(x)$ and actual targets $y$, for examples $(x, y) \sim P$. Then, we minimize the average of the loss function $\ell$ over the data distribution $P$, also known as the *expected risk*:

$$\boxed{R(f) = \int \ell(f(x), y) \mathrm{d}P(x, y).}$$

Unfortunately, the distribution $P$ is unknown in most practical situations. Instead, we usually have access to a set of training data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, where $(x_i, y_i) \sim P$ for all $i = 1, \ldots, n$. Using the training data $\mathcal{D}$, we may approximate $P$ by the *empirical distribution*

$$P_\delta(x, y) = \frac{1}{n} \sum_{i=1}^{n} \delta(x = x_i, y = y_i),$$

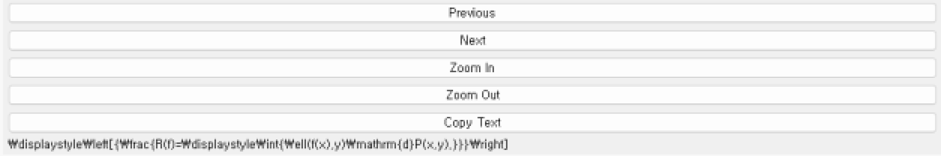
where $\delta(x = x_i, y = y_i)$ is a Dirac mass centered at $(x_i, y_i)$. Using the empirical distribution $P_\delta$, we can now approximate the expected risk by the *empirical risk*:

$$R_\delta(f) = \int \ell(f(x), y) \mathrm{d}P_\delta(x, y) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i). \quad (1)$$

```
₩displaystyle₩left[{₩frac{R(f)=₩displaystyle₩int{₩ell(f(x),y)₩mathrm{d}P(x,y),}}}₩right]
```

▶▶

## 변환된 Latex 언어 복사하여 사용



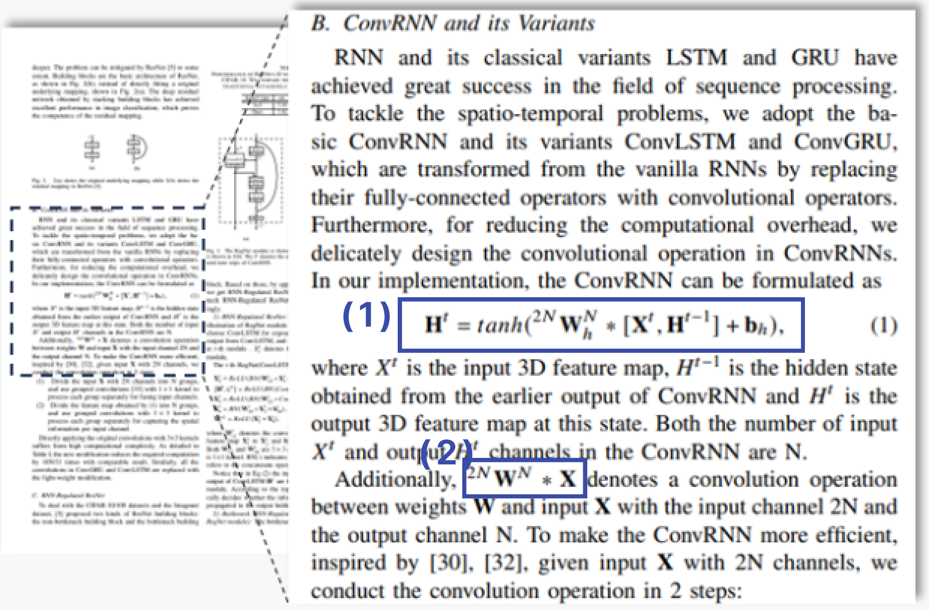
```
₩displaystyle₩left[{₩frac{R(f)=₩displaystyle₩int{₩ell(f(x),y)₩mathrm{d}P(x,y),}}}₩right]
```

# Method2 - 논문 속 모든 Formula를 Latex로 변환 후 csv로 반환

## Input

RegNet: Self-Regulated Network for Image Classification

## Formula Detection



*B. ConvRNN and its Variants*

RNN and its classical variants LSTM and GRU have achieved great success in the field of sequence processing. To tackle the spatio-temporal problems, we adopt the basic ConvRNN and its variants ConvLSTM and ConvGRU, which are transformed from the vanilla RNNs by replacing their fully-connected operators with convolutional operators. Furthermore, for reducing the computational overhead, we delicately design the convolutional operation in ConvRNNs. In our implementation, the ConvRNN can be formulated as

(1) $$\boxed{\mathbf{H}^t = tanh({}^{2N}\mathbf{W}_h^N * [\mathbf{X}^t, \mathbf{H}^{t-1}] + \mathbf{b}_h),} \quad (1)$$

where $X^t$ is the input 3D feature map, $H^{t-1}$ is the hidden state obtained from the earlier output of ConvRNN and $H^t$ is the output 3D feature map at this state. Both the number of input $X^t$ and output (2) $H^t$ channels in the ConvRNN are N.

Additionally, $\boxed{{}^{2N}\mathbf{W}^N * \mathbf{X}}$ denotes a convolution operation between weights **W** and input **X** with the input channel 2N and the output channel N. To make the ConvRNN more efficient, inspired by [30], [32], given input **X** with 2N channels, we conduct the convolution operation in 2 steps:

(1)

$$\mathbf{H}^t = tanh({}^{2N}\mathbf{W}_h^N * [\mathbf{X}^t, \mathbf{H}^{t-1}] + \mathbf{b}_h),$$

(2)

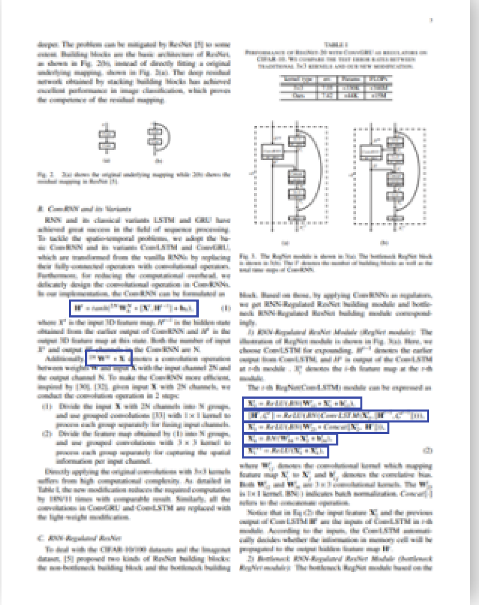
$${}^{2N}\mathbf{W}^N * \mathbf{X}$$

## Image To Latex

```
{₩bf H}^{t}=t a n h({}^{2N}{₩bf W}_{h}^{N}₩ast[{₩bf X}^{t},{₩bf H}^{t-1}]+{₩bf b}_{h}),

^{2N}W^{N} * X
```

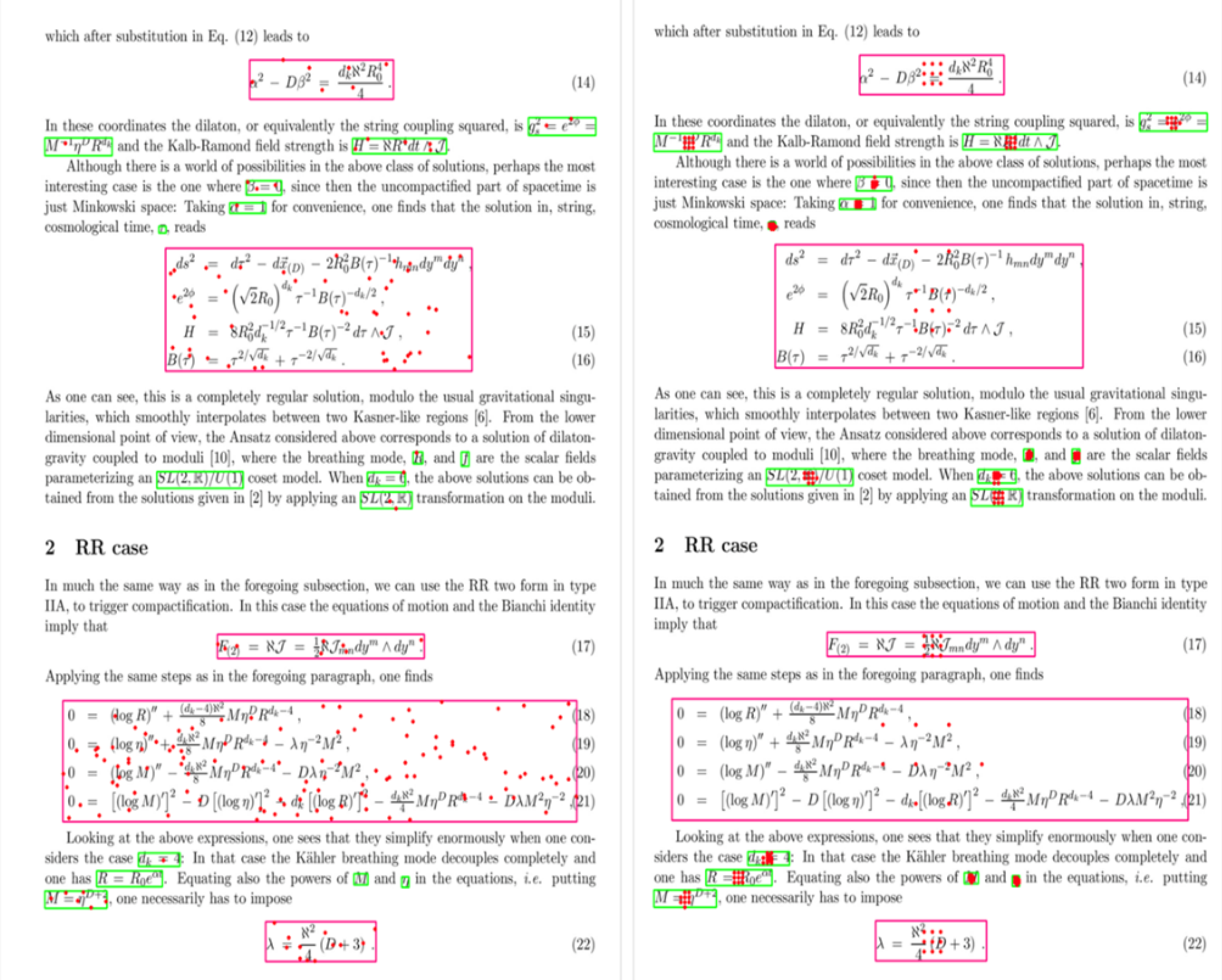
## Output



| No | Latex |
|---|---|
| 1 | {₩bf H}^{t}=t a n h({}^{2N}{₩bf W}_{h}^{N}₩ast[{₩bf X}^{t},{₩bf H}^{t-1}]+{₩bf b}_{h}), |
| 2 | ^{2N}W^{N} * X |

# Task1 - Formula Detection

문서, 이미지 or 스캔된 페이지에서 수학 공식 or 수학적 표현을 자동으로 위치



which after substitution in Eq. (12) leads to

$$\alpha^2 - D\beta^2 = \frac{d_k\aleph^2R_0^4}{4}. \tag{14}$$

In these coordinates the dilaton, or equivalently the string coupling squared, is $g_s^2 = e^{2\phi} = M^{-1}\eta^D R^{d_k}$ and the Kalb-Ramond field strength is $H = \aleph R^\alpha dt \wedge \mathcal{J}$.

Although there is a world of possibilities in the above class of solutions, perhaps the most interesting case is the one where $\beta = 0$, since then the uncompactified part of spacetime is just Minkowski space: Taking $\alpha = 1$ for convenience, one finds that the solution in, string, cosmological time, $\tau$, reads

$$\begin{aligned} ds^2 &= d\tau^2 - d\vec{x}_{(D)} - 2R_0^2 B(\tau)^{-1} h_{mn} dy^m dy^n, \\ e^{2\phi} &= \left(\sqrt{2}R_0\right)^{d_k} \tau^{-1} B(\tau)^{-d_k/2}, \\ H &= 8R_0^2 d_k^{-1/2} \tau^{-1} B(\tau)^{-2}\, d\tau \wedge \mathcal{J}, \quad (15) \\ B(\tau) &= \tau^{2/\sqrt{d_k}} + \tau^{-2/\sqrt{d_k}}. \quad (16) \end{aligned}$$

As one can see, this is a completely regular solution, modulo the usual gravitational singularities, which smoothly interpolates between two Kasner-like regions [6]. From the lower dimensional point of view, the Ansatz considered above corresponds to a solution of dilaton-gravity coupled to moduli [10], where the breathing mode, $B$, and $\mathcal{J}$ are the scalar fields parameterizing an $SL(2,\mathbb{R})/U(1)$ coset model. When $d_k = 0$, the above solutions can be obtained from the solutions given in [2] by applying an $SL(2,\mathbb{R})$ transformation on the moduli.

## 2 RR case

In much the same way as in the foregoing subsection, we can use the RR two form in type IIA, to trigger compactification. In this case the equations of motion and the Bianchi identity imply that

$$F_{(2)} = \aleph\mathcal{J} = \tfrac{1}{2}\aleph\mathcal{J}_{mn} dy^m \wedge dy^n. \tag{17}$$

Applying the same steps as in the foregoing paragraph, one finds

$$\begin{aligned} 0 &= (\log R)'' + \tfrac{(d_k-4)\aleph^2}{8} M\eta^D R^{d_k-4}, & (18) \\ 0 &= (\log \eta)'' + \tfrac{d_k\aleph^2}{8} M\eta^D R^{d_k-4} - \lambda\eta^{-2}M^2, & (19) \\ 0 &= (\log M)'' - \tfrac{d_k\aleph^2}{8} M\eta^D R^{d_k-4} - D\lambda\eta^{-2}M^2, & (20) \\ 0 &= \left[(\log M)'\right]^2 - D\left[(\log \eta)'\right]^2 - d_k\left[(\log R)'\right]^2 - \tfrac{d_k\aleph^2}{4} M\eta^D R^{d_k-4} - D\lambda M^2\eta^{-2}. & (21) \end{aligned}$$

Looking at the above expressions, one sees that they simplify enormously when one considers the case $d_k = 4$: In that case the Kähler breathing mode decouples completely and one has $R = R_0e^{\alpha t}$. Equating also the powers of $M$ and $\eta$ in the equations, *i.e.* putting $M = \eta^{D+2}$, one necessarily has to impose

$$\lambda = \frac{\aleph^2}{4}(D+3). \tag{22}$$

which after substitution in Eq. (12) leads to

$$\alpha^2 - D\beta^2 = \frac{d_k\aleph^2R_0^4}{4}. \tag{14}$$

In these coordinates the dilaton, or equivalently the string coupling squared, is $g_s^2 = e^{2\phi} = M^{-1}\eta^D R^{d_k}$ and the Kalb-Ramond field strength is $H = \aleph R^\alpha dt \wedge \mathcal{J}$.

Although there is a world of possibilities in the above class of solutions, perhaps the most interesting case is the one where $\beta = 0$, since then the uncompactified part of spacetime is just Minkowski space: Taking $\alpha = 1$ for convenience, one finds that the solution in, string, cosmological time, $\tau$, reads

$$\begin{aligned} ds^2 &= d\tau^2 - d\vec{x}_{(D)} - 2R_0^2 B(\tau)^{-1} h_{mn} dy^m dy^n, \\ e^{2\phi} &= \left(\sqrt{2}R_0\right)^{d_k} \tau^{-1} B(\tau)^{-d_k/2}, \\ H &= 8R_0^2 d_k^{-1/2} \tau^{-1} B(\tau)^{-2}\, d\tau \wedge \mathcal{J}, \quad (15) \\ B(\tau) &= \tau^{2/\sqrt{d_k}} + \tau^{-2/\sqrt{d_k}}. \quad (16) \end{aligned}$$

As one can see, this is a completely regular solution, modulo the usual gravitational singularities, which smoothly interpolates between two Kasner-like regions [6]. From the lower dimensional point of view, the Ansatz considered above corresponds to a solution of dilaton-gravity coupled to moduli [10], where the breathing mode, $B$, and $\mathcal{J}$ are the scalar fields parameterizing an $SL(2,\mathbb{R})/U(1)$ coset model. When $d_k = 0$, the above solutions can be obtained from the solutions given in [2] by applying an $SL(2,\mathbb{R})$ transformation on the moduli.

2 RR case

In much the same way as in the foregoing subsection, we can use the RR two form in type IIA, to trigger compactification. In this case the equations of motion and the Bianchi identity imply that

$$F_{(2)} = \aleph\mathcal{J} = \tfrac{1}{2}\aleph\mathcal{J}_{mn} dy^m \wedge dy^n. \tag{17}$$

Applying the same steps as in the foregoing paragraph, one finds

$$\begin{aligned} 0 &= (\log R)'' + \tfrac{(d_k-4)\aleph^2}{8} M\eta^D R^{d_k-4}, & (18) \\ 0 &= (\log \eta)'' + \tfrac{d_k\aleph^2}{8} M\eta^D R^{d_k-4} - \lambda\eta^{-2}M^2, & (19) \\ 0 &= (\log M)'' - \tfrac{d_k\aleph^2}{8} M\eta^D R^{d_k-4} - D\lambda\eta^{-2}M^2, & (20) \\ 0 &= \left[(\log M)'\right]^2 - D\left[(\log \eta)'\right]^2 - d_k\left[(\log R)'\right]^2 - \tfrac{d_k\aleph^2}{4} M\eta^D R^{d_k-4} - D\lambda M^2\eta^{-2}. & (21) \end{aligned}$$

Looking at the above expressions, one sees that they simplify enormously when one considers the case $d_k = 4$: In that case the Kähler breathing mode decouples completely and one has $R = R_0e^{\alpha t}$. Equating also the powers of $M$ and $\eta$ in the equations, *i.e.* putting $M = \eta^{D+2}$, one necessarily has to impose

$$\lambda = \frac{\aleph^2}{4}(D+3). \tag{22}$$

### 1. 텍스트 감지

→ 문서의 텍스트 영역을 감지

### 2. 공식 구별

→ 감지된 텍스트 영역 중에서 수학 공식 or 수식을 포함하고 있는 영역을 구별

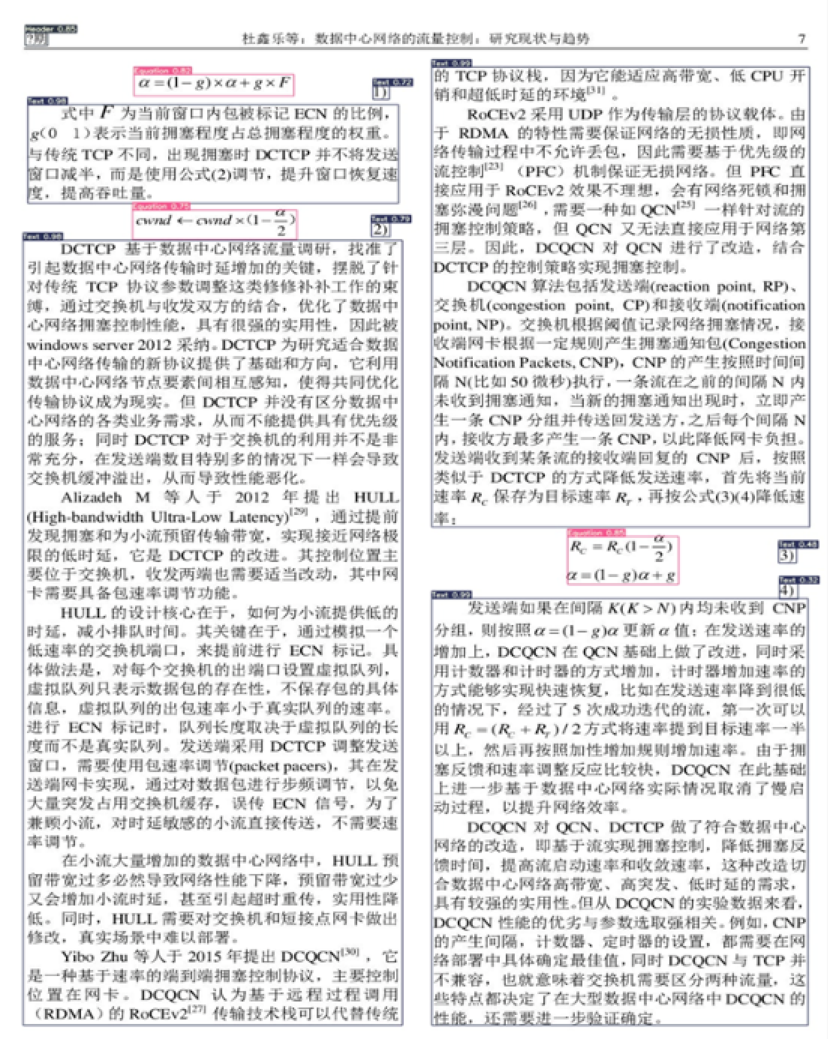
### 3. 공식 위치 파악

→ 수학 공식의 정확한 위치와 경계 파악

# Task1 - Formula Detection

## Model

CnSTD (Chinese Scene Text Detector )



杜鑫乐等：数据中心网络的流量控制：研究现状与趋势 7

$$\alpha=(1-g)\times\alpha+g\times F \quad (1)$$

式中 $F$ 为当前窗口内包被标记 ECN 的比例，$g$（0 1）表示当前拥塞程度占总拥塞程度的权重。与传统 TCP 不同，出现拥塞时 DCTCP 并不将发送窗口减半，而是使用公式(2)调节，提升窗口恢复速度，提高吞吐量。

$$cwnd\leftarrow cwnd\times(1-\frac{\alpha}{2}) \quad (2)$$

DCTCP 基于数据中心网络流量调研，找准了引起数据中心网络传输时延增加的关键，摆脱了针对传统 TCP 协议参数调整这类修修补补工作的束缚，通过交换机与收发双方的结合，优化了数据中心网络拥塞控制性能，具有很强的实用性，因此被 windows server 2012 采纳。DCTCP 为研究适合数据中心网络传输的新协议提供了基础和方向，它利用数据中心网络节点要素间相互感知，使得共同优化传输协议成为现实。但 DCTCP 并没有区分数据中心网络的各类业务需求，从而不能提供具有优先级的服务；同时 DCTCP 对于交换机的利用并不是非常充分，在发送端数目特别多的情况下一样会导致交换机缓冲溢出，从而导致性能恶化。

Alizadeh M 等人于 2012 年提出 HULL (High-bandwidth Ultra-Low Latency)[29]，通过提前发现拥塞和为小流预留传输带宽，实现接近网络极限的低时延，它是 DCTCP 的改进。其控制位置主要位于交换机，收发两端也需要适当改动，其中网卡需要具备包速率调节功能。

HULL 的设计核心在于，如何为小流提供低的时延，减小排队时间。其关键在于，通过模拟一个低速率的交换机端口，来提前进行 ECN 标记。具体做法是，对每个交换机的出端口设置虚拟队列，虚拟队列只表示数据包的存在性，不保存包的具体信息，虚拟队列的出包速率小于真实队列的速率。进行 ECN 标记时，队列长度取决于虚拟队列的长度而不是真实队列。发送端采用 DCTCP 调整发送窗口，需要使用包速率调节(packet pacers)，其在发送端网卡实现，通过对数据包进行步频调节，以免大量突发占用交换机缓存，误传 ECN 信号，为了兼顾小流，对时延敏感的小流直接传送，不需要速率调节。

在小流大量增加的数据中心网络中，HULL 预留带宽过多必然导致网络性能下降，预留带宽过少又会增加小流时延，甚至引起超时重传，实用性降低。同时，HULL 需要对交换机和短接点网卡做出修改，真实场景中难以部署。

Yibo Zhu 等人于 2015 年提出 DCQCN[30]，它是一种基于速率的端到端拥塞控制协议，主要控制位置在网卡。DCQCN 认为基于远程过程调用（RDMA）的 RoCEv2[27] 传输技术栈可以代替传统的 TCP 协议栈，因为它能适应高带宽、低 CPU 开销和超低时延的环境[31]。

RoCEv2 采用 UDP 作为传输层的协议载体。由于 RDMA 的特性需要保证网络的无损性质，即网络传输过程中不允许丢包，因此需要基于优先级的流控制[23]（PFC）机制保证无损网络。但 PFC 直接应用于 RoCEv2 效果不理想，会有网络死锁和拥塞弥漫问题[26]，需要一种如 QCN[25] 一样针对流的拥塞控制策略，但 QCN 又无法直接应用于网络第三层。因此，DCQCN 对 QCN 进行了改造，结合 DCTCP 的控制策略实现拥塞控制。

DCQCN 算法包括发送端(reaction point, RP)、交换机(congestion point, CP)和接收端(notification point, NP)。交换机根据阈值记录网络拥塞情况，接收端网卡根据一定规则产生拥塞通知包(Congestion Notification Packets, CNP)，CNP 的产生按照时间间隔 N(比如 50 微秒)执行，一条流在之前的间隔 N 内未收到拥塞通知，当新的拥塞通知出现时，立即产生一条 CNP 分组并传送回发送方，之后每个间隔 N 内，接收方最多产生一条 CNP，以此降低网卡负担。发送端收到某条流的接收端回复的 CNP 后，按照类似于 DCTCP 的方式降低发送速率，首先将当前速率 $R_c$ 保存为目标速率 $R_T$，再按公式(3)(4)降低速率：

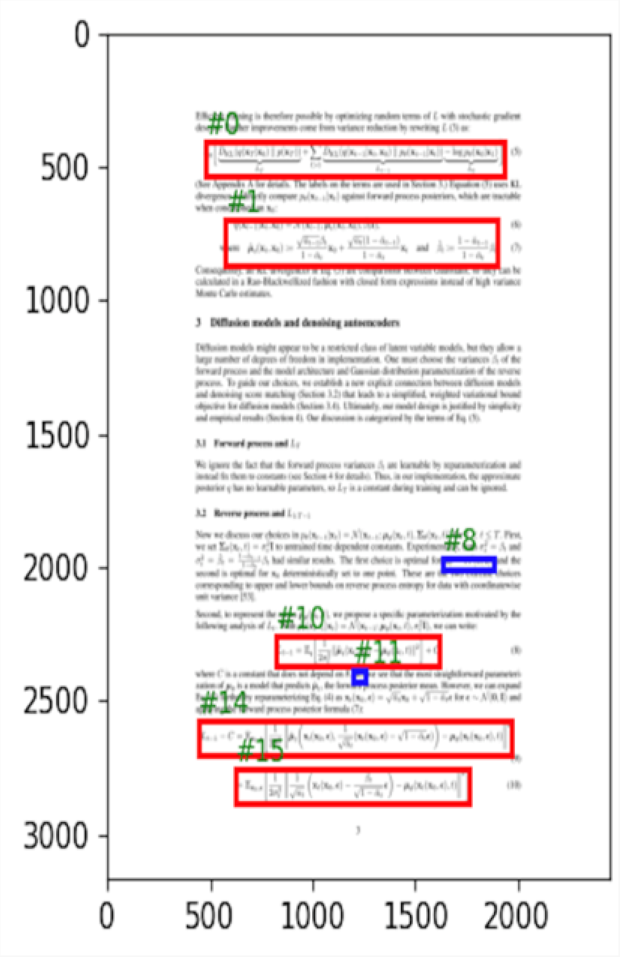
$$R_C=R_C(1-\frac{\alpha}{2}) \quad (3)$$

$$\alpha=(1-g)\alpha+g \quad (4)$$

发送端如果在间隔 $K(K>N)$ 内均未收到 CNP 分组，则按照 $\alpha=(1-g)\alpha$ 更新 $\alpha$ 值；在发送速率的增加上，DCQCN 在 QCN 基础上做了改进，同时采用计数器和计时器的方式增加，计时器增加速率的方式能够实现快速恢复，比如在发送速率降到很低的情况下，经过了 5 次成功迭代的流，第一次可以用 $R_C=(R_C+R_T)/2$ 方式将速率提到目标速率一半以上，然后再按照加性增加规则增加速率。由于拥塞反馈和速率调整反应比较快，DCQCN 在此基础上进一步基于数据中心网络实际情况取消了慢启动过程，以提升网络效率。

DCQCN 对 QCN、DCTCP 做了符合数据中心网络的改造，即基于流实现拥塞控制，降低拥塞反馈时间，提高流启动速率和收敛速率，这种改造切合数据中心网络高带宽、高突发、低时延的需求，具有较强的实用性。但从 DCQCN 的实验数据来看，DCQCN 性能的优劣与参数选取强相关。例如，CNP 的产生间隔，计数器、定时器的设置，都需要在网络部署中具体确定最佳值，同时 DCQCN 与 TCP 并不兼容，也就意味着交换机需要区分两种流量，这些特点都决定了在大型数据中心网络中 DCQCN 的性能，还需要进一步验证确定。

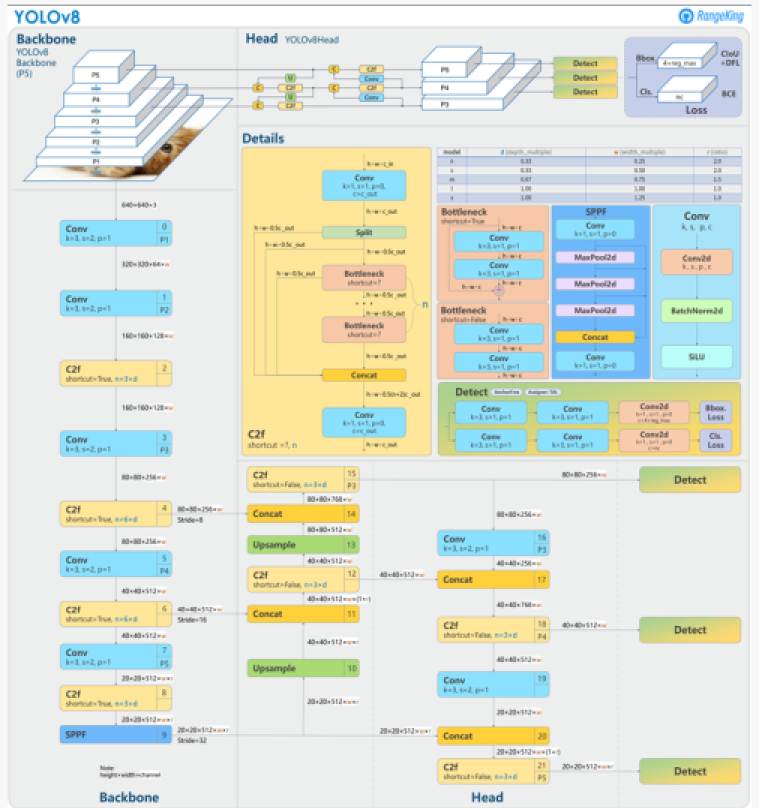
Inference



# Task1 - Formula Detection

## Model

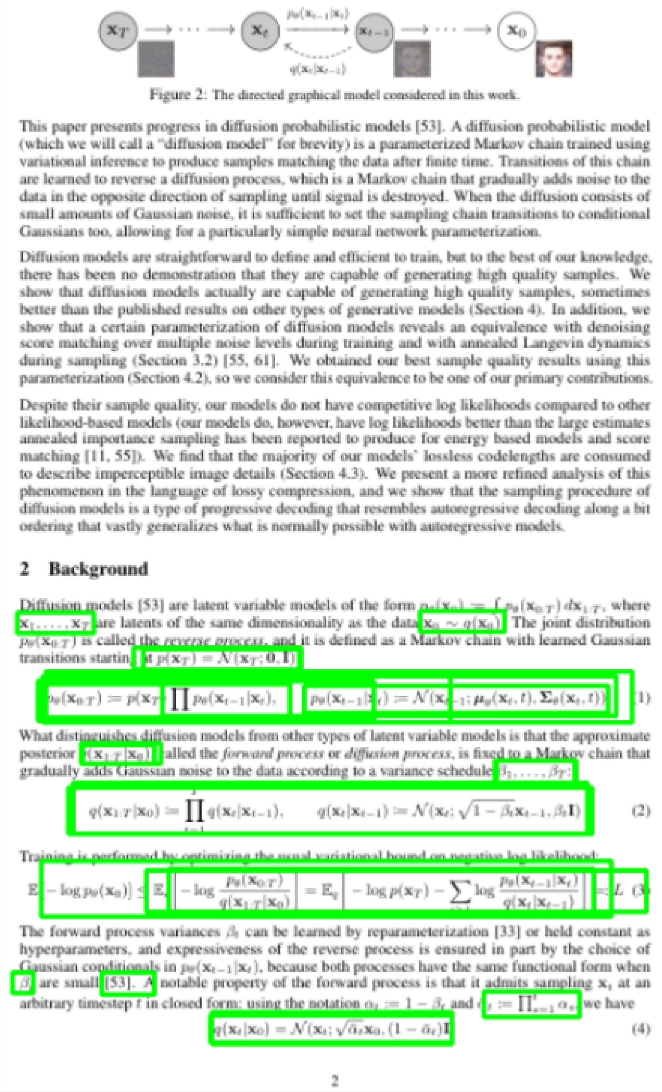
### YOLO v8



YOLO 모델을 위한 완전히 새로운 리포지토리를 출시하여 개체 감지, 인스턴스 세분화 및 이미지 분류 모델을 train하기 위한 통합 프레임워크로 구축

- YOLOv8n, YOLOv8x, YOLOv8m 등 다양한 버전이 존재
- YOLOv8m (medium)은 COCO에서 50.2의 mAP 달성

### Inference



$\mathbf{x}_T$
$\mathbf{x}_t$
$\mathbf{x}_{t-1}$
$\mathbf{x}_0$
$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$
$q(\mathbf{x}_t|\mathbf{x}_{t-1})$

Figure 2: The directed graphical model considered in this work.

This paper presents progress in diffusion probabilistic models [53]. A diffusion probabilistic model (which we will call a "diffusion model" for brevity) is a parameterized Markov chain trained using variational inference to produce samples matching the data after finite time. Transitions of this chain are learned to reverse a diffusion process, which is a Markov chain that gradually adds noise to the data in the opposite direction of sampling until signal is destroyed. When the diffusion consists of small amounts of Gaussian noise, it is sufficient to set the sampling chain transitions to conditional Gaussians too, allowing for a particularly simple neural network parameterization.

Diffusion models are straightforward to define and efficient to train, but to the best of our knowledge, there has been no demonstration that they are capable of generating high quality samples. We show that diffusion models actually are capable of generating high quality samples, sometimes better than the published results on other types of generative models (Section 4). In addition, we show that a certain parameterization of diffusion models reveals an equivalence with denoising score matching over multiple noise levels during training and with annealed Langevin dynamics during sampling (Section 3.2) [55, 61]. We obtained our best sample quality results using this parameterization (Section 4.2), so we consider this equivalence to be one of our primary contributions.

Despite their sample quality, our models do not have competitive log likelihoods compared to other likelihood-based models (our models do, however, have log likelihoods better than the large estimates annealed importance sampling has been reported to produce for energy based models and score matching [11, 55]). We find that the majority of our models' lossless codelengths are consumed to describe imperceptible image details (Section 4.3). We present a more refined analysis of this phenomenon in the language of lossy compression, and we show that the sampling procedure of diffusion models is a type of progressive decoding that resembles autoregressive decoding along a bit ordering that vastly generalizes what is normally possible with autoregressive models.

#### 2 Background

Diffusion models [53] are latent variable models of the form $p_\theta(\mathbf{x}_0) := \int p_\theta(\mathbf{x}_{0:T})\,d\mathbf{x}_{1:T}$, where $\mathbf{x}_1, \ldots, \mathbf{x}_T$ are latents of the same dimensionality as the data $\mathbf{x}_0 \sim q(\mathbf{x}_0)$. The joint distribution $p_\theta(\mathbf{x}_{0:T})$ is called the *reverse process*, and it is defined as a Markov chain with learned Gaussian transitions starting at $p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$:

$$p_\theta(\mathbf{x}_{0:T}) := p(\mathbf{x}_T)\prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t), \qquad p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) := \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)) \tag{1}$$

What distinguishes diffusion models from other types of latent variable models is that the approximate posterior $q(\mathbf{x}_{1:T}|\mathbf{x}_0)$, called the *forward process* or *diffusion process*, is fixed to a Markov chain that gradually adds Gaussian noise to the data according to a variance schedule $\beta_1, \ldots, \beta_T$:

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) := \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}), \qquad q(\mathbf{x}_t|\mathbf{x}_{t-1}) := \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}) \tag{2}$$

Training is performed by optimizing the usual variational bound on negative log likelihood:

$$\mathbb{E}\left[-\log p_\theta(\mathbf{x}_0)\right] \le \mathbb{E}_q\left[-\log\frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] = \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \sum_{t\ge 1}\log\frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] =: L \tag{3}$$

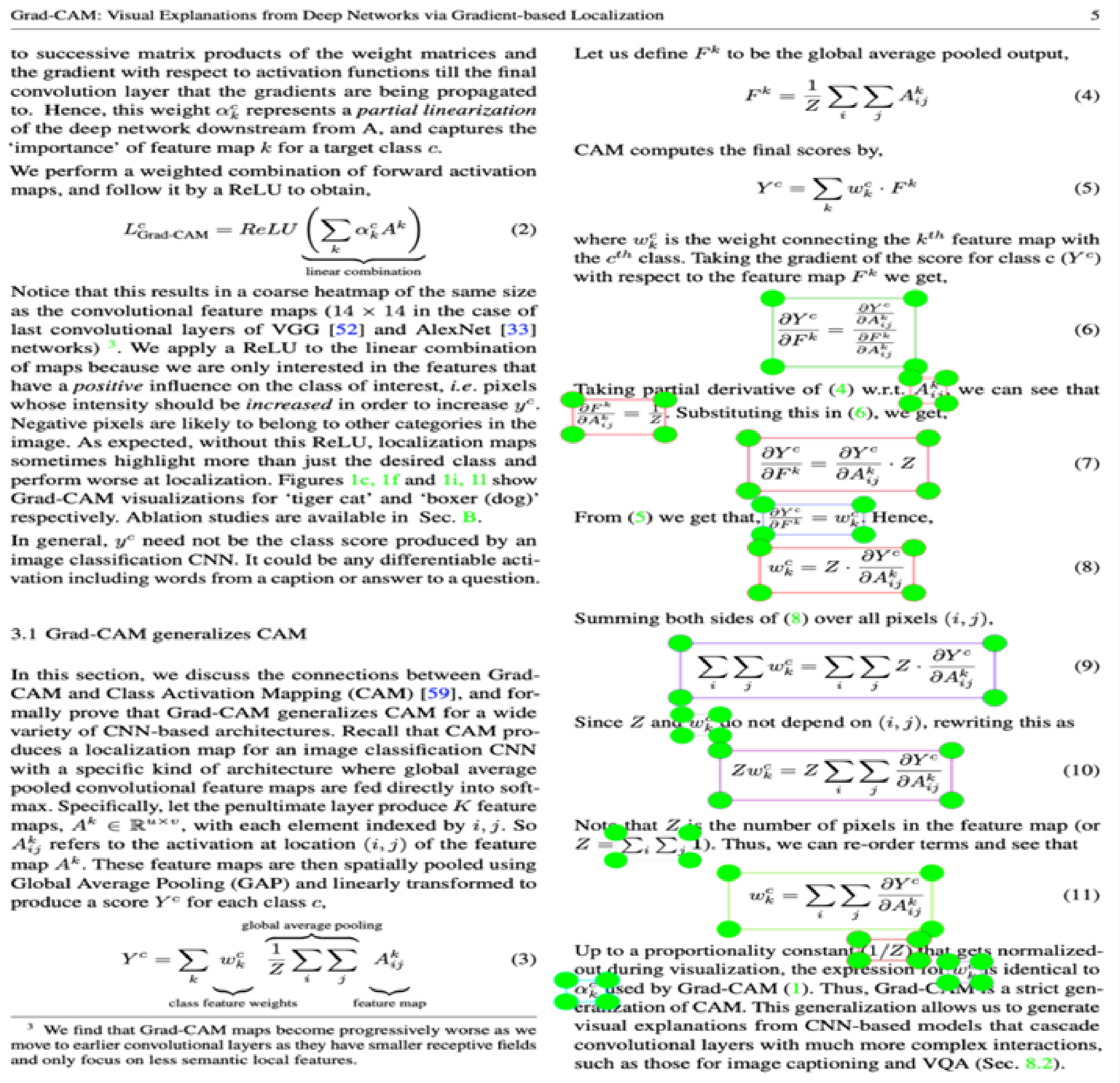
The forward process variances $\beta_t$ can be learned by reparameterization [33] or held constant as hyperparameters, and expressiveness of the reverse process is ensured in part by the choice of Gaussian conditionals in $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$, because both processes have the same functional form when $\beta_t$ are small [53]. A notable property of the forward process is that it admits sampling $\mathbf{x}_t$ at an arbitrary timestep $t$ in closed form: using the notation $\alpha_t := 1-\beta_t$ and $\bar{\alpha}_t := \prod_{s=1}^{t}\alpha_s$, we have

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}) \tag{4}$$

2

# Task1 - Formula Detection

## Data 생성



Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization 5

to successive matrix products of the weight matrices and the gradient with respect to activation functions till the final convolution layer that the gradients are being propagated to. Hence, this weight $\alpha_k^c$ represents a *partial linearization* of the deep network downstream from A, and captures the 'importance' of feature map $k$ for a target class $c$.

We perform a weighted combination of forward activation maps, and follow it by a ReLU to obtain,

$$L^c_{\text{Grad-CAM}} = ReLU \underbrace{\left( \sum_k \alpha_k^c A^k \right)}_{\text{linear combination}} \quad (2)$$

Notice that this results in a coarse heatmap of the same size as the convolutional feature maps (14 × 14 in the case of last convolutional layers of VGG [52] and AlexNet [33] networks) [3]. We apply a ReLU to the linear combination of maps because we are only interested in the features that have a *positive* influence on the class of interest, *i.e*. pixels whose intensity should be *increased* in order to increase $y^c$. Negative pixels are likely to belong to other categories in the image. As expected, without this ReLU, localization maps sometimes highlight more than just the desired class and perform worse at localization. Figures 1c, 1f and 1i, 1l show Grad-CAM visualizations for 'tiger cat' and 'boxer (dog)' respectively. Ablation studies are available in Sec. B.

In general, $y^c$ need not be the class score produced by an image classification CNN. It could be any differentiable activation including words from a caption or answer to a question.

### 3.1 Grad-CAM generalizes CAM

In this section, we discuss the connections between Grad-CAM and Class Activation Mapping (CAM) [59], and formally prove that Grad-CAM generalizes CAM for a wide variety of CNN-based architectures. Recall that CAM produces a localization map for an image classification CNN with a specific kind of architecture where global average pooled convolutional feature maps are fed directly into softmax. Specifically, let the penultimate layer produce $K$ feature maps, $A^k \in \mathbb{R}^{u \times v}$, with each element indexed by $i, j$. So $A^k_{ij}$ refers to the activation at location $(i, j)$ of the feature map $A^k$. These feature maps are then spatially pooled using Global Average Pooling (GAP) and linearly transformed to produce a score $Y^c$ for each class $c$,

$$Y^c = \sum_k \underbrace{w_k^c}_{\text{class feature weights}} \overbrace{\frac{1}{Z} \sum_i \sum_j}^{\text{global average pooling}} \underbrace{A^k_{ij}}_{\text{feature map}} \quad (3)$$

Let us define $F^k$ to be the global average pooled output,

$$F^k = \frac{1}{Z} \sum_i \sum_j A^k_{ij} \quad (4)$$

CAM computes the final scores by,

$$Y^c = \sum_k w_k^c \cdot F^k \quad (5)$$

where $w_k^c$ is the weight connecting the $k^{th}$ feature map with the $c^{th}$ class. Taking the gradient of the score for class c ($Y^c$) with respect to the feature map $F^k$ we get,

$$\frac{\partial Y^c}{\partial F^k} = \frac{\frac{\partial Y^c}{\partial A^k_{ij}}}{\frac{\partial F^k}{\partial A^k_{ij}}} \quad (6)$$

Taking partial derivative of (4) w.r.t. $A^k_{ij}$, we can see that $\frac{\partial F^k}{\partial A^k_{ij}} = \frac{1}{Z}$. Substituting this in (6), we get,

$$\frac{\partial Y^c}{\partial F^k} = \frac{\partial Y^c}{\partial A^k_{ij}} \cdot Z \quad (7)$$

From (5) we get that, $\frac{\partial Y^c}{\partial F^k} = w_k^c$. Hence,

$$w_k^c = Z \cdot \frac{\partial Y^c}{\partial A^k_{ij}} \quad (8)$$

Summing both sides of (8) over all pixels $(i, j)$,

$$\sum_i \sum_j w_k^c = \sum_i \sum_j Z \cdot \frac{\partial Y^c}{\partial A^k_{ij}} \quad (9)$$

Since $Z$ and $w_k^c$ do not depend on $(i, j)$, rewriting this as

$$Z w_k^c = Z \sum_i \sum_j \frac{\partial Y^c}{\partial A^k_{ij}} \quad (10)$$

Note that $Z$ is the number of pixels in the feature map (or $Z = \sum_i \sum_j 1$). Thus, we can re-order terms and see that

$$w_k^c = \sum_i \sum_j \frac{\partial Y^c}{\partial A^k_{ij}} \quad (11)$$

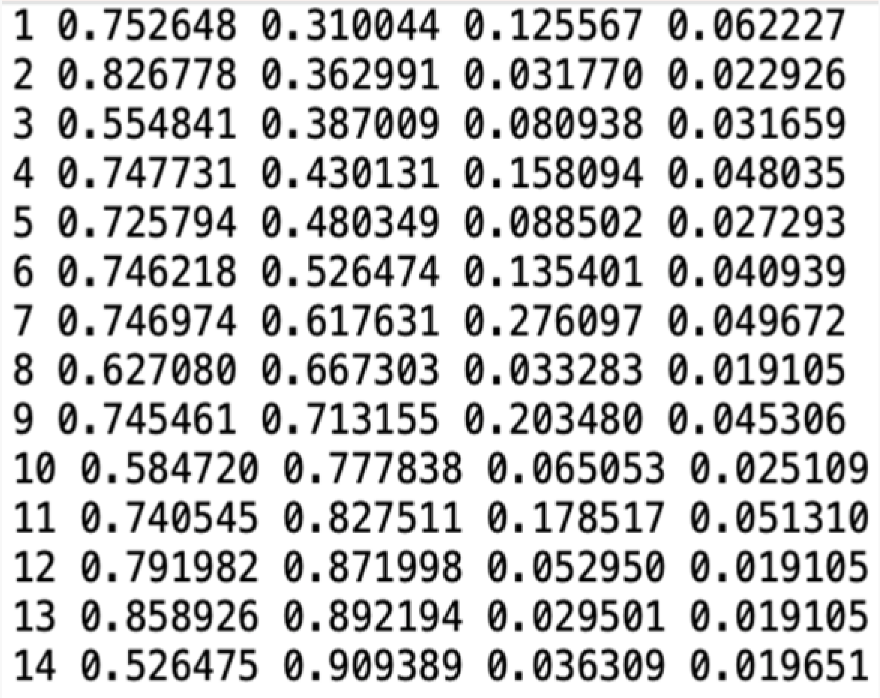
Up to a proportionality constant $(1/Z)$ that gets normalized-out during visualization, the expression for $w_k^c$ is identical to $\alpha_k^c$ used by Grad-CAM (1). Thus, Grad-CAM is a strict generalization of CAM. This generalization allows us to generate visual explanations from CNN-based models that cascade convolutional layers with much more complex interactions, such as those for image captioning and VQA (Sec. 8.2).

[3] We find that Grad-CAM maps become progressively worse as we move to earlier convolutional layers as they have smaller receptive fields and only focus on less semantic local features.

8개의 논문 이용

▶▶

**Labeling**

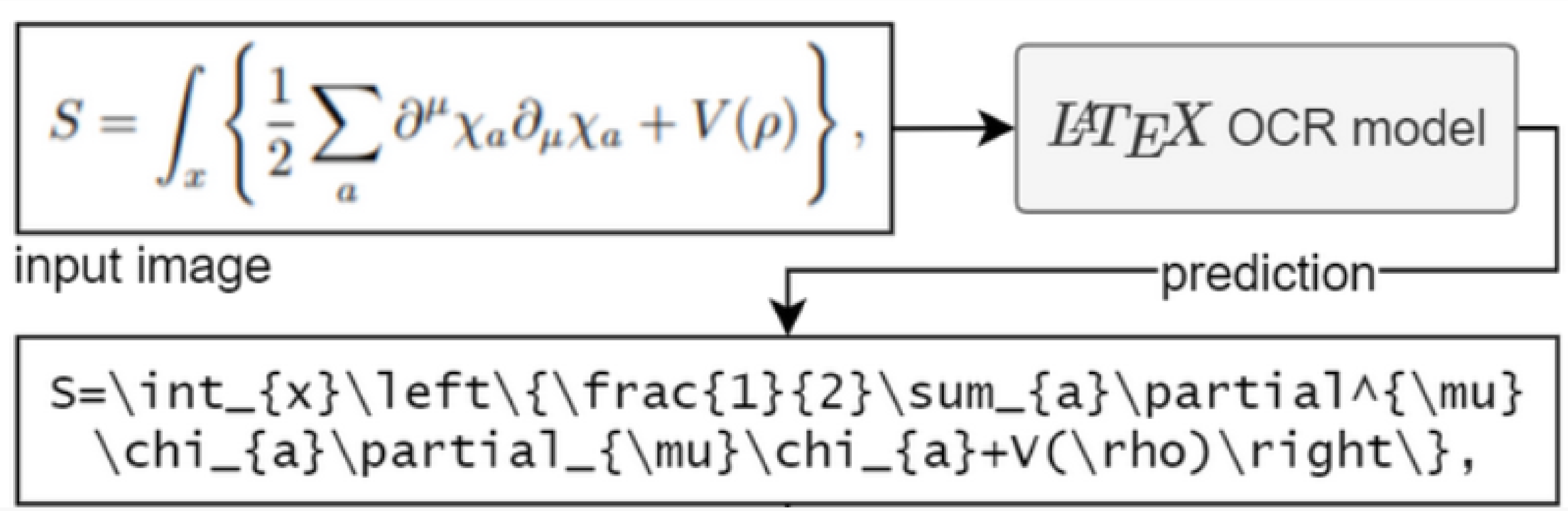


```
1 0.752648 0.310044 0.125567 0.062227
2 0.826778 0.362991 0.031770 0.022926
3 0.554841 0.387009 0.080938 0.031659
4 0.747731 0.430131 0.158094 0.048035
5 0.725794 0.480349 0.088502 0.027293
6 0.746218 0.526474 0.135401 0.040939
7 0.746974 0.617631 0.276097 0.049672
8 0.627080 0.667303 0.033283 0.019105
9 0.745461 0.713155 0.203480 0.045306
10 0.584720 0.777838 0.065053 0.025109
11 0.740545 0.827511 0.178517 0.051310
12 0.791982 0.871998 0.052950 0.019105
13 0.858926 0.892194 0.029501 0.019105
14 0.526475 0.909389 0.036309 0.019651
```

→ 학습까지 완료

# Task2 - Formula To Latex

수학 공식을 LaTeX 형식으로 변환하는 도구나 알고리즘

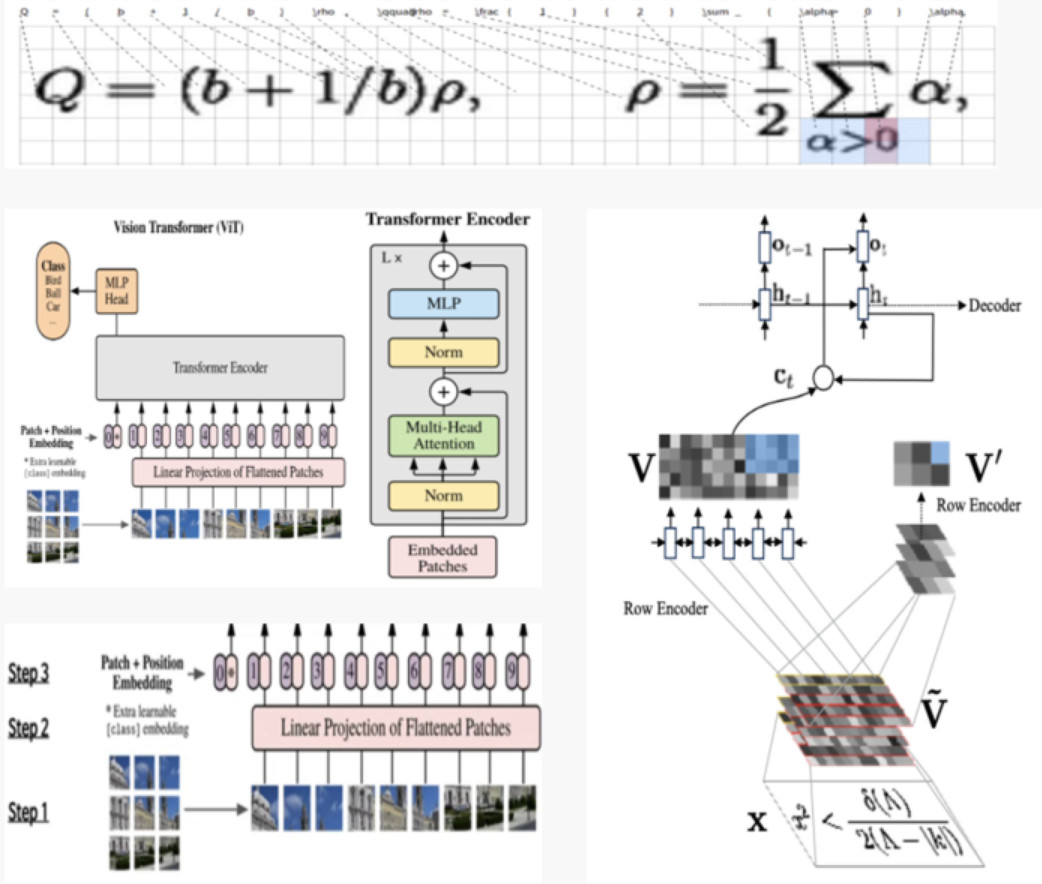

$S=\int_{x}\left\{\frac{1}{2}\sum_{a}\partial^{\mu}\chi_{a}\partial_{\mu}\chi_{a}+V(\rho)\right\},$
input image
LaTeX OCR model
prediction

```
S=\int_{x}\left\{\frac{1}{2}\sum_{a}\partial^{\mu}
\chi_{a}\partial_{\mu}\chi_{a}+V(\rho)\right\},
```

# Task2 - Formula To Latex

## Model

### pix2tex - LatexOCR



$Q=(b+1/b)\rho, \qquad \rho=\frac{1}{2}\sum_{\alpha>0}\alpha,$
Vision Transformer (ViT)
Transformer Encoder
MLP Head
Class
Bird
Ball
Car
...
Transformer Encoder
Patch + Position Embedding
* Extra learnable [class] embedding
Linear Projection of Flattened Patches
L ×
MLP
Norm
Multi-Head Attention
Norm
Embedded Patches
Step 3
Step 2
Step 1
Patch + Position Embedding
* Extra learnable [class] embedding
Linear Projection of Flattened Patches
Decoder
Row Encoder
Row Encoder
V
V'
Ṽ
X

### Inference

$$\Psi=\exp\left(il\varphi\right)R\left(\rho\right).$$

$$\xi_n(u)=\sin\left(\frac{\pi nu}{t}\right) \qquad n=1,2,\dots$$

$$\delta^V(t)g=g((t^2-1)\dot{X}(t)X(t)^{-1}+I),$$

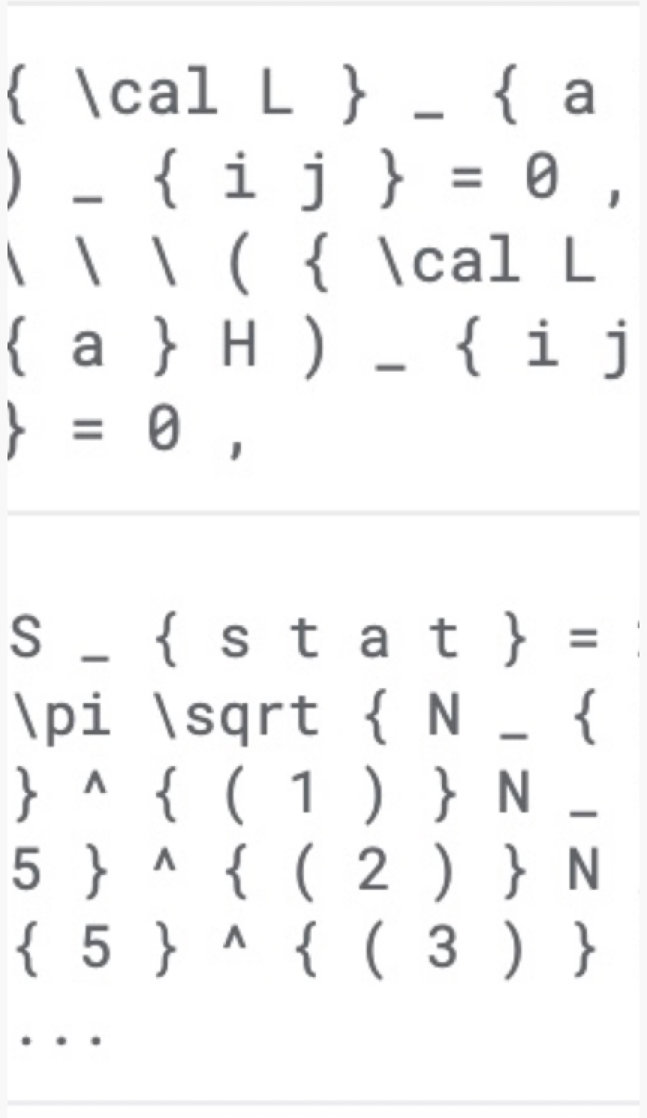
$$\partial_{\bar{z}}^2H_F+H_p\partial_{\bar{z}}^2H_F=0, \quad \partial_{\bar{z}}^2H_p=0.$$

| | |
|---|---|
| 5abbb9b19f. png | $\int_{-c}^{\infty}dl\,e^{-l\zeta}\int_{-c}^{\infty}dl'e^{-l'\zeta}ll'\frac{l'-l}{l+l'}\{3\,\delta''(l)-\frac{3}{4}t\,\delta(l)\}=0.$ |
| 329a44c373.png | $[K_a^-(p),K_b^-(k)]=if_{abc}K_c^-(p+k)-2\pi^2Cp\delta_{ab}\delta_{p+k,0}.$ |
| 73b51f198b. png | $E(v)=\frac{d}{dt}E(q) \qquad \forall t\,.$ |
| 6331d9e7fd. png | $\frac{1}{L^2}\prod_{i=1}^{3}(r_+^2+q_i)-\mu r_+^2\sim 0\,.$ |
| 91a55d2cb9.png | $x^I(\sigma+2\pi,\tau)=x^I(\sigma,\tau)+2\pi L^I.$ |
| 408fe63a30.png | $\xi_\sigma(\sigma)=1-\frac{\pi e^\sigma}{3}+8\pi e^\sigma\sum_{n=1}^{\infty}\frac{ne^{-2n\pi e^\sigma}}{1-e^{-2n\pi e^\sigma}}$ |
| 232d6fea7c. png | $m_0^2\varphi+\frac{\mu^2-m_0^2}{\beta}\tan(\beta\varphi)=0$ |
| 702a6a1e50.png | $\frac{dz}{dr}=e^{2A}, \qquad \psi\to e^{-3A/2}\psi$ |
| 7a7df95199.png | $F_{A1}\,(q^{a1},\dot{q}^{a1})=0\,,\ F_{A0}\,(q^{a1})=0\,.$ |
| d39642fa83.png | $1-\frac{\delta\phi}{2\pi}=L'(\infty)=\frac{2GW+1}{N^2(\infty)}.$ |
| 2c65d65f05.png | $\varphi''+\frac{1}{r}\varphi'+\lambda\left(\varphi-\varphi^3\right)=0,$ |
| 3bea2f425a. png | $x^N+y^N=z^N,\ \ \Phi(l)=\omega^{l(l+N)/2},\ \ \omega^{1/2}=\exp(\pi i/N).$ |
| 3e3f5d7c1f. png | $e_{\hat{\mu}}^{\hat{\mu}}=\begin{pmatrix}e_\sigma^s & 0\\ 0 & e_\alpha^a\end{pmatrix}$ |
| 12e9e7dd1f. png | $\phi=\begin{pmatrix}0\\ \phi_0\end{pmatrix}$ |
| 7c99ded238.png | $\mathrm{Tr}(\mathbf{M}_+\mathbf{M}_-)=0\,.$ |
| 652ec3e732.png | $\int_{C_t}B_2=4\pi^2\alpha'\frac{1}{2}\equiv b_0$ |

# Task2 - Formula To Latex

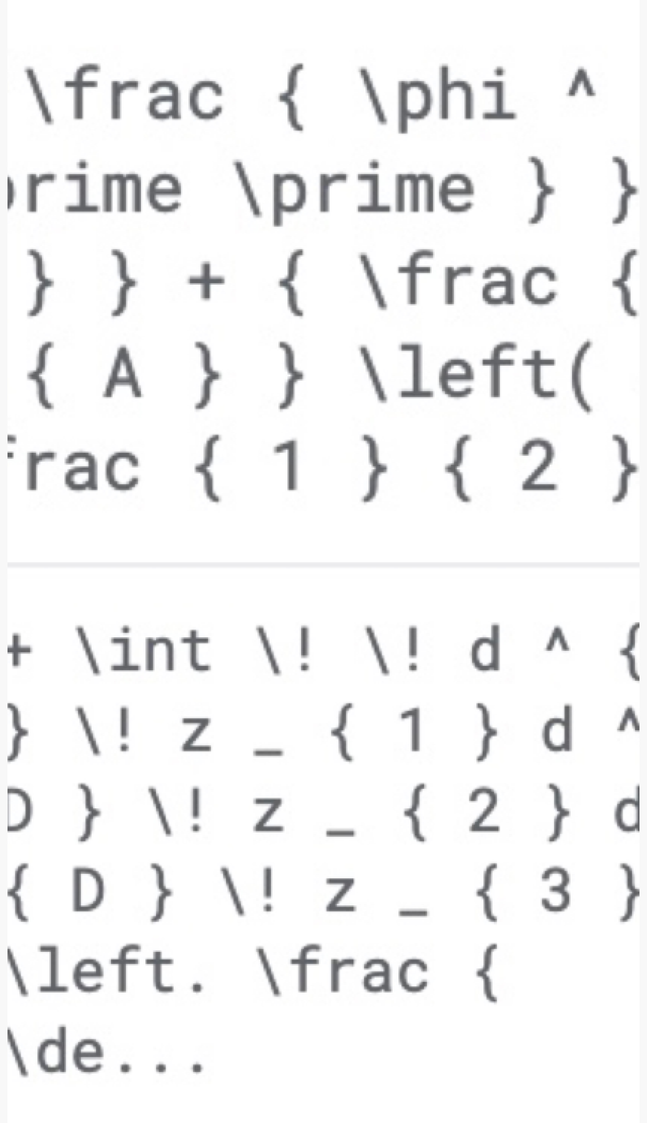
## Data

M2LATEX-100K (Kaggle)



```
{ \cal L } _ { a
) _ { i j } = 0 ,
\ \ \ ( { \cal L
{ a } H ) _ { i j
} = 0 ,
```

```
S _ { s t a t } =
\pi \sqrt { N _ {
} ^ { ( 1 ) } N _
5 } ^ { ( 2 ) } N
{ 5 } ^ { ( 3 ) }
...
```



```
\frac { \phi ^
rime \prime } }
} } + { \frac {
{ A } } \left(
rac { 1 } { 2 }
```

```
+ \int \! \! d ^ {
} \! z _ { 1 } d ^
D } \! z _ { 2 } d
{ D } \! z _ { 3 }
\left. \frac {
\de...
```

# **WebUI** Inference

# Formula Extract Inference

# 기대효과

논문 리뷰 작성할 때
수식 입력이 편해짐

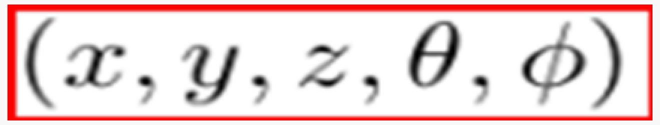
처음보는 수학 기호에
대한 Latex를
찾아 볼 필요가 없음

전체 논문 번역 시
자연스러운 번역 결과가
나올 것으로 기대

# 결론

## 아쉬운 점

- WebUI에서 드래그 하는 영역의 빨간 박스가 Latex모델에 같이 전달되는 문제
- WebUI를 예쁘게 만들지 못함
- 대량의 데이터를 활용하지 못함



$(x, y, z, \theta, \phi)$

## 앞으로의 계획

- 학습된 YOLO 모델 적용
- TTS -> Latex 언어로 변환된 Formula를 Speech로 읽는 법을 알려줌
- Formula를 Latex로 변환하여 논문 전체를 한번에 Translation
- 배포