格式要求：英文字体Times New Roman，表示变量的字体斜体，数字正体，符号格式采用symbol，符号前后空格，中文字体宋体或微软雅黑，首行缩进两字符，行间距1.5倍，两端对齐，word正文字体10号。

----------------------------------------分割线----------------------------------------

内容模版(以下为正文)：

**基于Transformer的机器翻译算法研究**

*Abstract*

1. Introduction

**Background and significance**

随着互联网快速发展，人们聊天不再局限于国内，各种社交软件的诞生使得与外国友人聊天的机会越来越多，一定程度上促进了对异国风情、文化的兴趣和学习。对于某国语言精通者，交流和学习并不是一件困难的事情；但是对于普通人来说不同语种可能导致传递信息时产生歧义，因此如Google翻译和Microsoft翻译等翻译软件逐渐问世，解决了多数人不同语言的沟通问题。早期这些翻译软件存在一定通病，如单纯翻译字符或词语，并没有考虑上下文、语序、时态等问题，这就导致了翻译结果与理想产生了一定程度的偏差。由此，科学家们希望借助计算机来解决这些问题，机器翻译的概念逐渐浮现在人们的眼中。

自然语言处理(NLP)是智能技术领域的一个重要分支，旨在让计算机理解、生成和处理人类生成的自然语言，机器翻译属于NLP的一个重要应用，旨在自动将源语言文本翻译成另一种目标语言文本的技术。其研究历史最早可以追溯到20世纪30年代。1933年，法国科学家G.B.阿尔楚尼提出了使用机器来完成自然语言翻译任务的想法。在1946年，世界上第一台现代电子计算机ENIAC诞生。随后不久，信息论的先驱、美国科学家Warren Weaver于1947年提出了计算机进行语言自动翻译的想法，紧接着在1949年发表《翻译备忘录》，里面正式提出机器翻译的概念和思想，后续学者们纷纷加入讨论和研究中。1954年，美国乔治敦大学在IBM公司协同下，使用IBM-701计算机首次完成了英语转俄语的机器翻译试验。这次实验中IBM-701计算机有史以来第一次自动将60个俄语句子翻译成了英语，但是没人提到这些翻译得到的样本是经过人工多重挑选和测试的，从而排除了歧义性。正当机器翻译的话题热火朝天时，1964年美国科学院成立了Auto Language Processing Advisory Committee，其经过两年的研究，于1966年11月公布了一份ALPAC报告，里面明确且全面否定了机器翻译的可行性，因此各大公司停止了机器翻译研究的资金支持，从而开始了大约11年的受挫时间，各公司的机器翻译项目数量锐减。

1970年中后期，计算机技术和语言学的快速发展，社会信息服务的需求提高，之前的限制条件被逐一打破，促进了机器翻译的研究复苏。随着计算机相关领域的发展，20世纪末的1993年，IBM公司的Brown和Della Pietra等人基于统计学首次提出了基于词对齐的翻译模型，这是一个重要的里程碑，标志着现代机器翻译方法的诞生。2003年，Franz Och基于机器学习思想提出对数线性模型及其权重训练方法，其发表的文章中提出了基于短语的翻译模型和最小错误率训练方法，标志着机器翻译的真正崛起。2006年，Google翻译作为一个免费服务正式发布，拥有连接互联网设备的人们纷纷享受到不同语言快速翻译的便利性，也为Google公司提供了更多翻译数据，便于展开更多的机器翻译研究工作。随着数据量的增加，出现许许多多的生僻词、超长句、特殊单词等传统机器学习算法无法较好解决的问题，如果增加算法的步骤和复杂度，计算机无法在短期内能够完成权重的训练。随着计算机硬件的发展，显卡GPU计算技术逐渐优化改良，这很好的解决了权重训练时间长的问题，学者们在算法、模型架构上能够更大胆地增加计算步骤以优化传统机器学习算法的缺陷。

2013年，Nal Kalchbrenner和Phil Blunsom提出了Recurrent Continuous Translation Models，这是一种用于机器翻译的新型End-to-end Encoder-Decoder Architecture，它使用卷积神经网络（CNN）将给定源文本编码为连续向量，紧接着使用循环神经网络（RNN）作为解码器，将状态向量转换为目标语言，这是一个重要的研究，后续出现的各种架构都是基于Encoder-Decoder Architecture的改良，如2014年Bengio提出的Sequence to sequence learning with neural networks，该网络架构与前者不同，其Encoder和Decoder均采用RNN结构的LSTM，并且实验表示该架构上线Google翻译后，翻译的质量已经超越了部分人类。在2015年Bahdanau D.等人提出在Encoder-Decoder结构中加入对其的Attention权重，Vaswani A.等人基于Attention权重的思想于2017年发表了一篇撼动机器翻译、乃至整个智能技术领域的论文，该论文中提出的Encoder-Decoder结构Sequence-to-sequence模型——Transformer，与以往的Sequence-to-sequence模型不同，该模型抛弃了一切的CNN和RNN，整个网络结构几乎都是由Attention机制组成，主要创新点在于Multi-Head、Self-Attention，这些设计使得该模型训练的收敛时间大大减少，并且实验结果表明，它的性能和鲁棒性远超当时所有模型架构。

本文将逐一剖析Transformer模型Encoder-Decoder架构，复现原文Transformer模型的各处细节和训练策略，在英语转德语和英语转法语的基础上，新增另外几种语言的数据集，验证复现模型的性能。

1. Related studies

**Basic ideas**

**Methods**

**Analysis of differences among them**

1. *Model Architecture*

Transformer和大多数competitive neural sequence transduction models相同，均为Encoder-Decoder架构，其整体架构如图1.

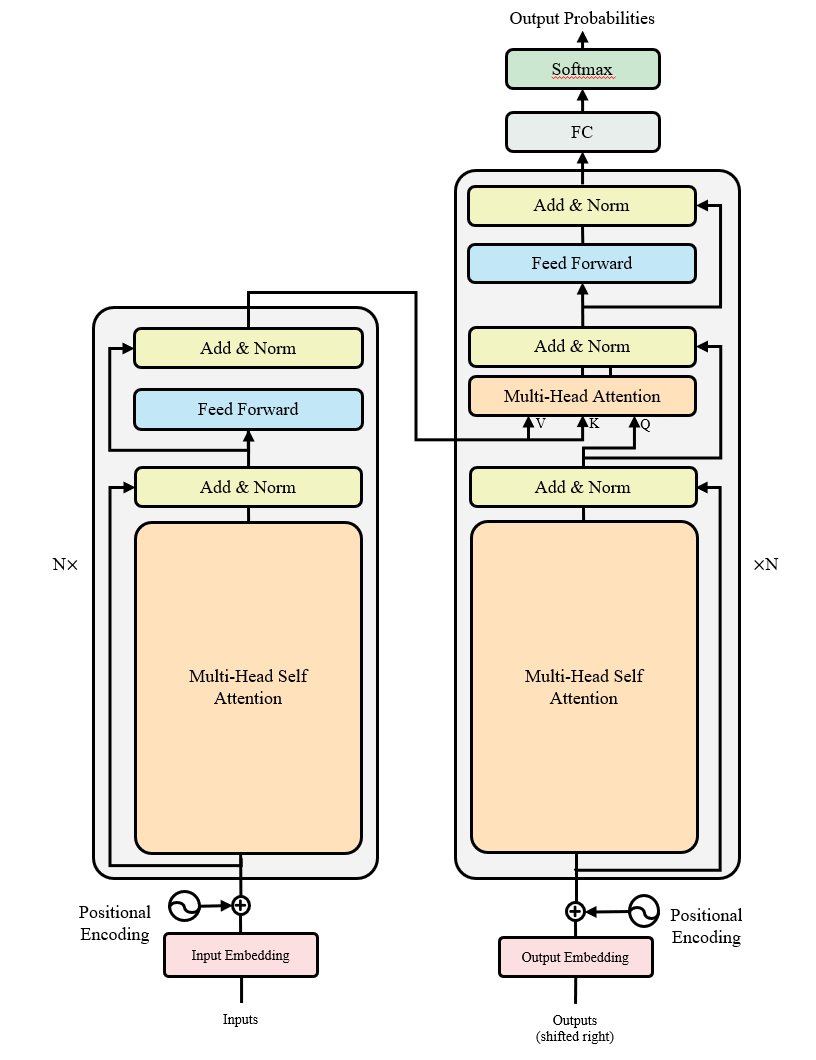


Figure 1. The Transformer model architecture.

其中，左侧为Encoder，左侧为Decoder，每个Encoder/Decoder都由个Encoder/Decoder blocks“夹心式”组成。青色方块FC表示Fully Connected Layer，也称Linear Layer；紫色小方块MatMul表示Matrix Multiplication；黄色小方块Concat表示矩阵连接；黄色方块Add & Norm表示残差连接和Layer Normalization (LN)。

1. Encoder/Decoder block

在整体架构上，Encoder和Decoder均是采取“夹心式“设计的个blocks组成。假设Encoder由个Encoder blocks组成，形状（表示句子长度，表示词向量维度）的词向量Inputs经过第一个Encoder block前向计算后，得到的outputs与inputs是相同形状的，即均为形状，详细结构如图2。

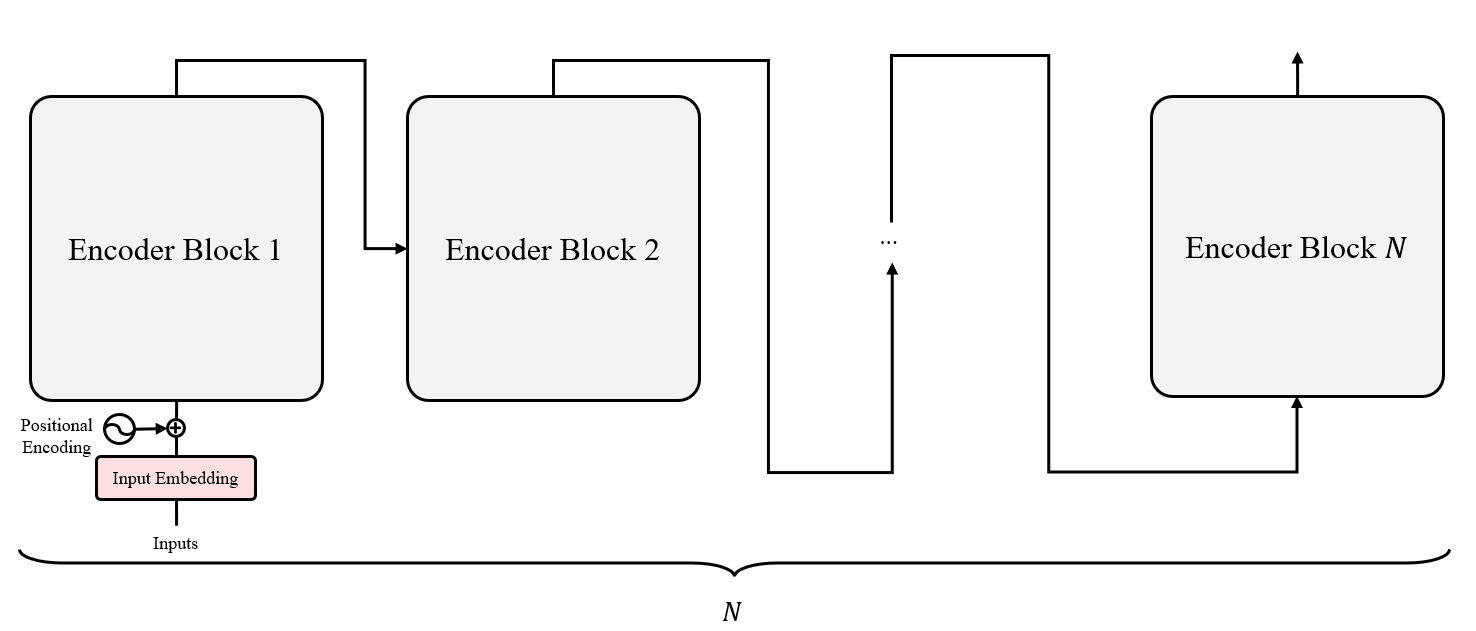


Figure 2. The Encoder forward calculation detailed architecture.

Decoder同理，但是其与Encoder block不同的是在Multi-head Self Attention (MHSA)后增加了一个MHA，用于接受Encoder block的输出和解码器内部的MHSA输出，从而完成注意力机制计算，详细结构如图3.

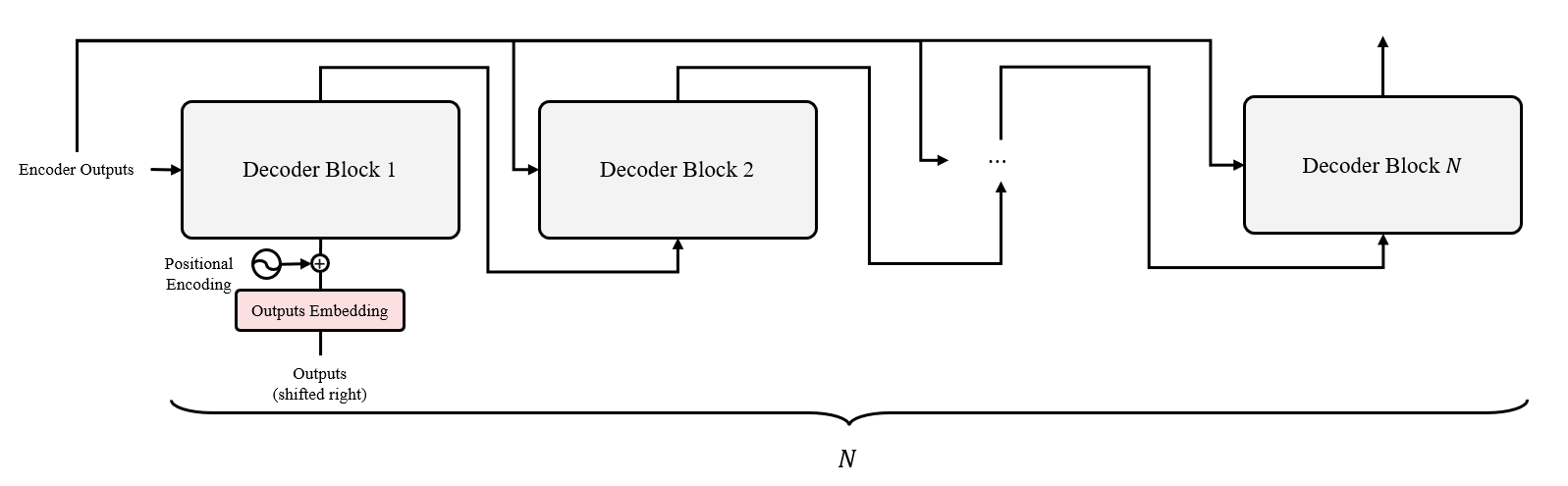


Figure 3. The Decoder forward calculation detailed architecture.

在训练过程中会要求输入形状的目标语言词向量进行训练，这里的shifted right表示剔除了标记句子起始的特殊单词。

1. *Multi-Head Attention*

上文提到的MHSA和MHA均是相同的计算过程，由于输入不同导致其计算的注意力矩阵具有不同的解释。原文中描述MHA由个Single Head Attention组成，与以往的注意力机制不同，其将inputs拆分成了份，经过个头的计算，最终进行Concat和FC层的映射得到与inputs相同的形状，其中单个头的注意力计算采用的是Scaled Dot-product Attention，MHA详细结构如图4.

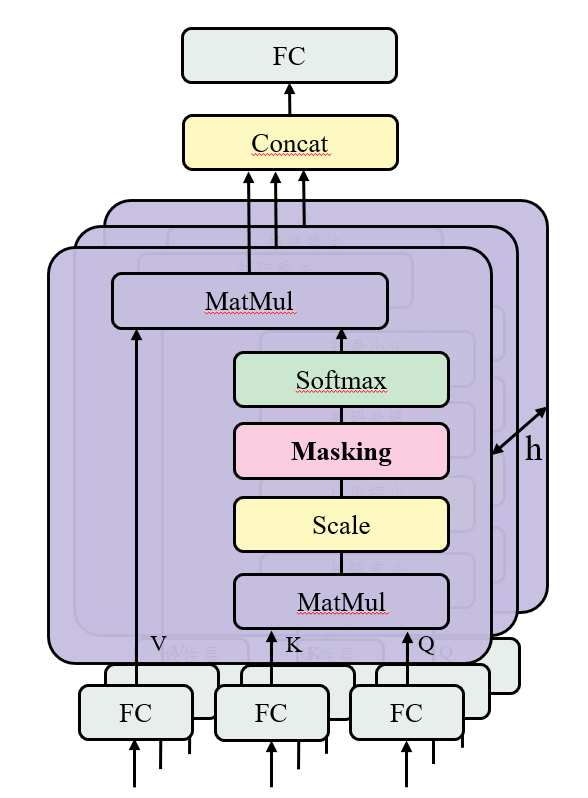


Figure 4. Multi-head Attention architecture.

在单个头的Scaled Dot-product Attention中，inputs经过三个FC层映射得到的分别表示“Query”、“Key”、“Value”，其中和的矩阵维度均为，的矩阵维度为。显而易见，MHSA的均来自inputs的复制，而MHA会出现差异，具体体现在Decoder block的MHSA的下一层MHA中，其和来自于Encoder outputs。计算公式如式(1)：

根据Fig 4.和式(1)，可以得知是一个方阵。在Encoder中，的形状是，由于和均是来源于inputs的FC映射，因此该方阵经过Scaled、Masking、Softmax等运算后可以表示inputs对自己的注意力权重矩阵，与 Matmul后即可得到一个自注意力“处理”后的词向量，一定程度上能够使得模型理解上下文关系，从而得到更优质的翻译结果。

在Attention的计算当中，涉及了一个至关重要的运算，便是Masking。在庞大的数据中，不可能保证每一条句子长短相同，因此在训练前会对数据进行padding操作，填充空白字符使得在当前批次下句子长度是相同的。而在Scaled Dot-product Attention中，计算注意力权重矩阵时，我们不希望空白字符对它自己产生更多“关注”，因此需要一个0-1二值化的Mask来“遮住”填充字符。Masking本质上是对填充字符位置的词向量加上一个无限小的数，因此在填充字符处对应的Mask值为True（即1），其余字符为False（即0），在True的位置上将会被累加一个无限小的数，源语言词向量的mask可视化如图5.

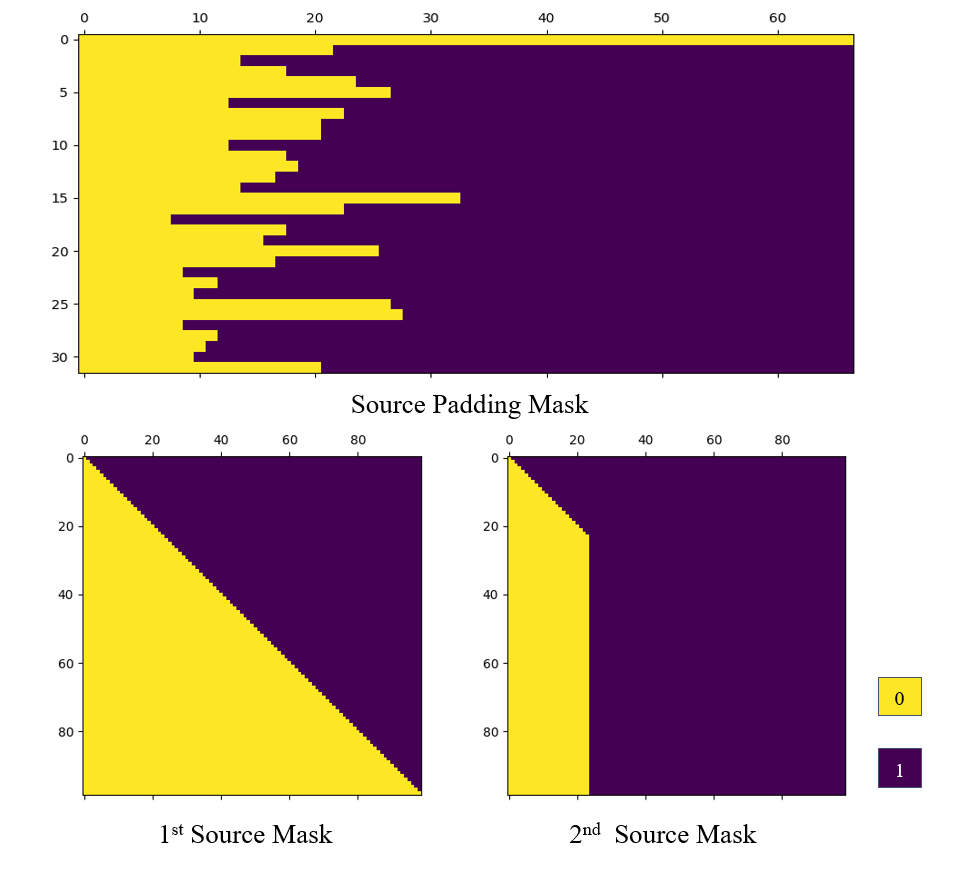


Figure 5. Visualzation of the generated source mask.

对于每一条句子，都会产生一个方阵mask。在批数据训练模式中，若再生成batch size数量的mask，无疑会增加GPU的运算消耗，因此根据Source/Target Padding Mask来调整最终的Source/Target Mask。

1. *Feed Forward Network*

在每一个Encoder/Decoder block中都会包含一个Feed Forward Network，主要是由两个线性变换和一个ReLU激活函数组成，其计算如式(2).

其中，，并且要远大于。原文中每一个提到Encoder/Decoder blocks的可能不相同，本文后续实验将统一是相同的。

1. *Add & Norm*

在每一个MHA或FFN后，都会进行一个Residual Add和Layer Normalization (LN)。Residual Add是Kaiming He等人于2015年发表的文章中提出模型结构中，将相同形状的inputs和outputs进行累加，一定程度上可以避免前向计算中特征的损失。

LN是一种较新的data normalization方式。由于机器翻译中不可能保证每条句子长度相同，对于长短不一的两条句子，同一位置的词向量分布大概率是不同的，另外Testing中可能会出现比Training最大长度还要长的句子，对于多余部分的词向量没有训练的参数可用，所以和CNN一样使用Batch Normalization是没有任何意义的；而CNN和Transformer中使用LN的方法不同，如图6。

**（缺少）**

Figure 6. Three normalization methods for visualization.

令输入的表示MHA或FFN的运算结果，由此可以得到Add & Norm的计算公式如式(3)。

1. *Positional Encoding*

一条句子是带有上下文关系的，但是经过Embedding后得到的词向量元素之间是相互独立的，没有办法能够得到具体的上下文信息。因此在进入Encoder/Decoder前，会进行Positional Encoding的累加，来使词向量得到位置特征。原文中提供了两种positional encoding的方法：

1. 根据人为经验公式得到，通常采取的是正余弦函数如下。

其中，表示矩阵中词索引，表示维度。即位置编码的每个维度对应于正弦曲线。波长形成从到的几何级数字。选择正余弦函数是因为假设它可以让模型轻松学习相对位置。

1. 使用Embedding层的权重与词向量做Matmul来生成最终输入的词向量。

第一种方法的好处是能够在短时间内使模型收敛，但是模型翻译质量的上限一定程度上取决于经验公式的选取；而第二种方法的好处是整体模型一旦收敛，其翻译质量的上限将远高于第一种，缺点是模型训练过程收敛困难，可能花很长一段时间都无法收敛。

1. *Loss function*

原文中并没有提及使用的损失函数，根据机器翻译任务的要求，其最后输出结果的处理相当于是一个多分类任务，在多分类任务上诸多学者采取的是Cross Entropy损失函数，如式(4)。

其中，表示类别总数量，在这里相当于目标语言的词典数量。由于在机器翻译数据中，对长短不一句子进行了Padding操作，因此填充字符是没必要计算损失的。

1. *Evaluation metric*

为了评估模型的好坏，因此需要一个评价指标。本文是采取了Accuracy和BLEU两种评价指标。使用Accuracy的原因是因为损失函数使用了Cross Entropy，因此希望添加多一个评价指标来衡量模型性能。

1. Analysis, discussion, and future work

**For example, advantages, disadvantages, and future work**

1. *Dataset*
2. *Data preprocessing and post-processing*
3. *Optimizer*
4. *Results*
5. Conclusions

Reference (Examples):

1. K. Shibata, "CIM Systems in An Advanced Semiconductor Factory. " *International Symposium on Semiconductor Manufacturing, Extended Abstracts of ISSM*, pp. 53-56, 1994.
2. G. D. Antonio, J. S. Bedolla, and P. Chiabert, "A Novel Methodology to Integrate Manufacturing Execution Systems with the Lean Manufacturing Approach," *Procedia Manufacturing*, vol. 11, pp. 2243-2251, 2017.
3. R. Beregi, G. Pedone, B. Háy, and J. Váncza, "Manufacturing Execution System Integration through the Standardization of a Common Service Model for Cyber-Physical Production Systems." *Applied Sciences*, vol.11, no. 16, pp.7581, 2021.
4. B. K. Choi and B. H. Kim, “MES (manufacturing execution system) architecture for FMS compatible to ERP (enterprise planning system)," *International Journal of Computer Integrated Manufacturing*, vol. 15, no. 3, pp. 274-284, 2002.

The citation format of references in the main body of the paper shown as follows:

* + - 1. [Shibata, 1995; Antonio *et al*., 2017];
      2. [Choi and Kim, 2002];